

■ PROJECTS ■ THEORY ■ APPLICATIONS ■ CIRCUITS ■ TECHNOLOGY

NUTS
AND
VOLTS

NUTS AND VOLTS

www.nutsvolts.com
April 2015

EVERYTHING FOR ELECTRONICS

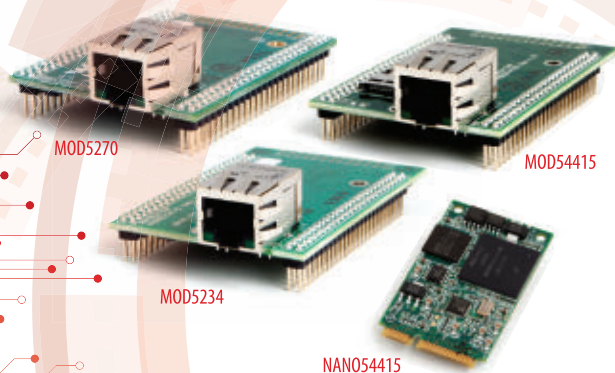
Organize Your Bench With

3D

Printed Storage

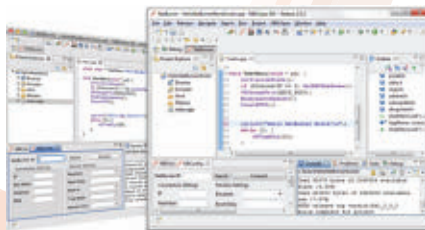


Ethernet Core Modules with High-Performance Connectivity Options



- **MOD5270**
147.5 MHz processor with 512KB Flash & 8MB RAM · 47 GPIO · 3 UARTs · I²C · SPI
- **MOD5234**
147.5 MHz processor with 2MB flash & 8MB RAM · 49 GPIO · 3 UARTs · I²C · SPI · CAN · eTPU (for I/O handling, serial communications, motor/timing/engine control applications)
- **MOD54415**
250 MHz processor with 32MB flash & 64MB RAM · 42 GPIO · 8 UARTs · 5 I²C · 3 SPI · 2 CAN · SSI · 8 ADC · 2 DAC · 8 PWM · 1-Wire® interface
- **NANO54415**
250 MHz processor with 8MB flash & 64MB RAM · 30 GPIO · 8 UARTs · 4 I²C · 3 SPI · 2 CAN · SSI · 6 ADC · 2 DAC · 8 PWM · 1-Wire® interface

Add Ethernet connectivity to an existing product, or use it as your product's core processor



The goal: Control, configure, or monitor a device using Ethernet

The method: Create and deploy applications from your Mac or Windows PC. Get hands-on familiarity with the NetBurner platform by studying, building, and modifying source code examples.

The result: Access device from the Internet or a local area network (LAN)

The NetBurner Ethernet Core Module is a device containing everything needed for design engineers to add network control and to monitor a company's communications assets. For a very low price point, this module solves the problem of network-enabling devices with 10/100 Ethernet, including those requiring digital, analog and serial control.

MOD5270-100IR.....\$69 (qty. 100)	NNDK-MOD5270LC-KIT\$99
MOD5234-100IR.....\$99 (qty. 100)	NNDK-MOD5234LC-KIT\$249
MOD54415-100IR.....\$89 (qty. 100)	NNDK-MOD54415LC-KIT\$129
NANO54415-200IR...\$69 (qty. 100)	NNDK-NANO54415-KIT.....\$99

NetBurner Development Kits are available to customize any aspect of operation including web pages, data filtering, or custom network applications. The kits include all the hardware and software you need to build your embedded application.

➤ **For additional information please visit**
<http://www.netburner.com/kits>

Build a robot rover in one evening

Then take it out for a test ride while the sun sets. See it zip around, equipped with a sense of sight, sound, smell, speed, heat or cold; or thunder detection, speech recognition... **whatever you imagine**, there's probably a sensor or transceiver click™ board that can do it (there's more than a 100 available). Just plug one in to add a function. Zero hardware setup. And you choose the MCU for the driver's seat. **Introducing the Buggy** – a dream car for makers and hackers.



April 2015

24 Build Your Own Arduino Barograph

Feeling pressure? Then, you need a barograph to measure it!

■ By Mark McGuire

28 Build the Super KISS Timer

Ever built a project that needed an accurate timer to switch something on or off at regular intervals? See if this simple one fits the bill.

■ By Frank Muratore

32 Build the Annoy-O-Matic

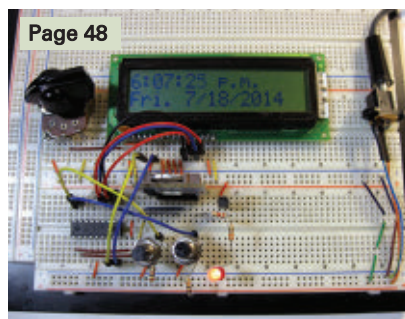
You'll always be ready for April Fool's Day with this easy circuit that will help bring practical jokes to a whole new level.

■ By Bob Diaz

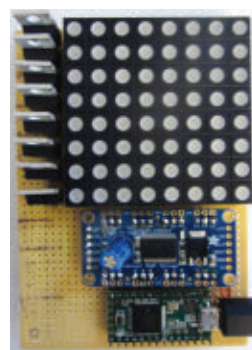
38 Beyond the Arduino — Part 2

Last month, we established a hardware platform to work on. This time, we'll dive right into working in a new environment.

■ By Andrew Retallack



Page 48



Page 52

48 Add a Real Time Clock to Your PIC Projects

RTCs have all kinds of neat uses in microcontroller projects. Get introduced to this wonderful world of timekeeping with the popular and inexpensive TinyRTC.

■ By Thomas Henry

52 Driving LEDs with a Microcontroller

Typically, one of the first experiments people do when working with microcontrollers is to blink an LED. However, the thrill of this wears out pretty quickly, so let's see what else can be done.

■ By Craig A. Lindley

Columns

08 TechKnowledgey 2015 Events, Advances, and News

The world's largest digital camera has been funded, Intel is getting sticky, you can get skin care via your iPhone, plus there's lots more to read about.

12 Q&A Reader Questions Answered Here

Making a smart house smarter, a telephone off-hook alarm, and radio tuning are question topics this time.

16 The Spin Zone Adventures in Propeller Programming

Can We Talk? Again?

This month's project goes through setting up a simple command language for controlling the Propeller through a serial interface.

58 The Design Cycle Advanced Techniques for Design Engineers

The RN4020 PICtail Plus BLE.

Bluetooth 4.0 is becoming the de facto standard when it comes to embedded device monitoring and control from iPhones and Android devices.

Microchip's BLE radio module is called the RN4020. Unlike many of the other BLE offerings, the RN4020 Bluetooth Low Energy radio module comes backed with a full load of example code and hardware development tools. This month, we'll get started on the BLE path with Microchip's RN4020 PICtail, a terminal emulator, and a laptop.

67 Practical 3D Printing Real World Uses for the Electronics Experimenter

3D Custom Storage Boxes.

Get your electronics über organized with specially designed storage drawers.

70 Near Space Approaching the Final Frontier

CubeSats — Part 3:

Attitude and Velocity.

Learn about some "toys" that help CubeSats control their own attitude and velocity.

Departments

06	DEVELOPING PERSPECTIVES	66	ELECTRO-NET
	Amateur Radio — Not Just for the Nostalgic	74	NV WEBSTORE
07	READER FEEDBACK	77	CLASSIFIEDS
22	NEW PRODUCTS	78	TECH FORUM
23	SHOWCASE	81	AD INDEX

GEARMOTORS

Put your project in motion!

\$24.99

3-12V Standard

\$34.99

6-12V Precision

NEW

\$27.99

3-12V Precision Planetary

NEW

\$39.99

6-12V Precision Planetary

\$9.99

6-12V Micro

NEW

3-12V Planetary Gearmotor
with Encoder! **\$49.99**

4mm D-shaft

- Dual Ball Bearings
- Full Metal Gears
- 3PPR Magnetic Encoder
- 11 different speeds

6-pin JST connector

Tons of compatible accessories

12V LINEAR ACTUATORS

\$129.99

12V Heavy Duty Linear Actuator
- offered in 6 stroke lengths: 2"-12"
- aluminum stroke rod & outer tube
- metal drive gears
- choose from 25, 115 or 180 lb thrust!

\$399.99

12V Super Duty Linear Actuator
- offered in 6 stroke lengths: 4"-24"
- stainless steel rod & outer tube
- precision ball screw drive

NEW

MOTOR MOUNTS

Our growing selection of aluminum motor mounts make it easy to find the perfect mount for your application!

NOMAD OFF-ROAD CHASSIS

\$279.99

Motor mount #555176 is used in conjunction with our 313 RPM Heavy Duty Planetary Gearmotors on the NOMAD 4WD chassis.

+ MOTOR CONTROLLERS -

Digital Manual Speed Controller

\$79.99

DMSC6-16-10

RoboClaw Motor Controllers

\$199.99

2x 60A

Multi mode interface can be controlled from standard RC radio, serial devices, analog and or microcontrollers such as an Arduino or Raspberry Pi.

www.SERVOCITY.com

Copyright 1999 - 2015 Robotzone, LLC - All rights reserved.
ServoCity™ and Actobotics™ are registered trademarks of Robotzone, LLC.

SAVE

Great educational discounts available. Apply today!

ENTER TO WIN

Fun contests with awesome prizes - like \$1,000 worth of parts!



by Bryan Bergeron, Editor

DEVELOPING PERSPECTIVES

Amateur Radio – Not Just for the Nostalgic

When I received my new catalog from MFJ Enterprises, I couldn't help but feel a twinge of nostalgia. Scanning through the pages of the latest ham gear revealed very little has changed since my youth. Sure, some of the instruments sport LCD displays instead of analog meters or LED displays, but from a gross technological perspective, the catalog could have been from the '70s. There was the usual mix of antennas, antenna rotors, linear amplifiers, antenna tuners, watt meters, microphones, and even a handful of iambic keyers for CW operation using Morse Code.

It's no secret that amateur radio has been in decline for a while, hastened by the popularity of the Internet. It used to be an accomplishment to chat with someone in Africa or Japan. Plus, slow-scan TV was good for perhaps a frame every couple seconds. Today, all that's required for world-wide video and audio communications is a cell phone – not a room packed with powerful gear. I can remember calibrating my wall clock and oscillator circuits with signals from WWV at 5.0 MHz and 10.0 MHz. Today, of course, clocks with built-in receivers update the displayed time automatically.

In short, most electronics enthusiasts don't consider amateur radio at the cusp of innovation in technology. And,

perhaps it isn't. However, if you're really serious about learning and experiencing electronics, you owe it to yourself to check out what amateur radio has to offer. I still use the diagnostic techniques I learned building and troubleshooting communications gear today – on both digital and analog circuits.

I still remember my first moonbounce communications using a microwave transceiver and an antenna array that automatically tracked the moon. Of course, it took months to prepare for what was about a minute of communications time. There was learning about high gain antenna arrays, and then using coat hangers and aluminum tubing to construct an array. There was working with waveguide and heliax, and figuring out the trajectory of the moon on a particular night. Plus, there were a couple dozen other problems that had to be solved. As a result, I learned a lot with each project. At least for me, it isn't about the final conversation, it's the process of building a system with specific capabilities and then operating it to the best of my ability.

You won't find that sort of challenge or excitement working on a bench with, say, a microcontroller and a few LEDs. It's one thing to build a power supply to use one day on your projects, and another to build one specifically to

power a transceiver that has to stand up to the rigors of emergency use. Amateur radio has a long history of public service. I spent many hurricane seasons in Louisiana providing communications for hundreds of families in temporary shelters. That's when knowing how to set up an antenna with duct tape and coat hangers paid off – not only for me, but for everyone in the shelters.

In future issues of *Nuts & Volts*, we're going to feature articles targeting the amateur radio community. If you're new to amateur radio, I invite you to at least skim the articles. I think you'll like what you find. If you happen to be a seasoned ham, then please consider contributing an article for your fellow *Nuts & Volts* readers.

73,
NU1N **NV**

Feedback Motion Control

The Old Way

- 1) Build robot
- 2) Guess PID coefficients
- 3) Test
- 3a) Express disappointment
- 3b) Search Internet, modify PID values
- 3c) Read book, modify PID coefficients again
- 3d) Decide performance is good enough
- 3e) Realize it isn't
- 3f) See if anyone just sells a giant servo
- 3g) Express disappointment
- 3h) Re-guess PID coefficients
- 3i) Switch processor
- 3j) Dust off old Differential Equations book
- 3k) Remember why the book was so dusty
- 3l) Calculate new, wildly different PID coefficients
- 3m) Invent new, wildly different swear words
- 3n) Research fuzzy logic
- 3o) Now it is certainly not working in uncertain ways
- 3p) Pull hair
- 3q) Switch controller
- 3r) Re-guess PID coefficients
- 3s) Switch programming language
- 3t) Start a new project that doesn't need feedback control
- 3u) See parts in box. Feel guilty. Go back to old project
- 3v) Start testing every possible combination of PID coefficients
- 3w) Apply eye drops to red, bleary, sleep-deprived eyes
- 3x) Wait. It's working!
- 3y) Decide not to do any more projects that require control systems.
- 3z) Wonder why someone doesn't just make a thing that tunes itself

The Kangaroo x2 Way

- 1) Build robot
- 2) Press Autotune
- 3) Get a snack

**Kangaroo x2
adds self-tuning
feedback to SyRen
and Sabertooth motor
drivers.**



\$24.99

DimensionEngineering

www.dimensionengineering.com/kangaroo

Published Monthly By

T & L Publications, Inc.

430 Princeland Ct.

Corona, CA 92879-1300

(951) 371-8497

FAX **(951) 371-3052**

Webstore orders only **1-800-783-4624**

www.nutsvolts.com

Subscription Orders

Toll Free **1-877-525-2539**

Outside US **1-818-487-4545**

P.O. Box 15277

North Hollywood, CA 91615

FOUNDER

Jack Lemieux

PUBLISHER

Larry Lemieux
publisher@nutsvolts.com

ASSOCIATE PUBLISHER/ ADVERTISING SALES

Robin Lemieux
robin@nutsvolts.com

EDITOR

Bryan Bergeron
techedit-nutsvolts@yahoo.com

VP OF OPERATIONS

Vern Graner
vern@nutsvolts.com

CONTRIBUTING EDITORS

Fred Eady	Tim Brown
Jon McPhalen	Jeff Eckert
Paul Verhage	Chuck Hellebuyck
Andrew Retallack	Thomas Henry
Craig Lindley	Frank Muratore
Mark McGuire	Bob Diaz

CIRCULATION DEPARTMENT

subscribe@nutsvolts.com

SHOW COORDINATOR

Audrey Lemieux

WEB CONTENT

Michael Kaudze
website@nutsvolts.com

WEBSTORE MARKETING

Brian Kirkpatrick
sales@nutsvolts.com

WEBSTORE MANAGER

Sean Lemieux

ADMINISTRATIVE STAFF

Debbie Stauffacher
Re Gandara

Copyright © 2015 by T & L Publications, Inc.
All Rights Reserved

All advertising is subject to publisher's approval. We are not responsible for mistakes, misprints, or typographical errors. *Nuts & Volts Magazine* assumes no responsibility for the availability or condition of advertised items or for the honesty of the advertiser. The publisher makes no claims for the legality of any item advertised in *Nuts & Volts*. This is the sole responsibility of the advertiser. Advertisers and their agencies agree to indemnify and protect the publisher from any and all claims, action, or expense arising from advertising placed in *Nuts & Volts*. Please send all editorial correspondence, UPS, overnight mail, and artwork to: 430 Princeland Court, Corona, CA 92879.

Printed in the USA on SFI & FSC stock.



READER FEEDBACK

Im-PART-ing Knowledge

I was wondering where J.W. Koebel gets his replacement parts from for his radio repairs.

Lee Gernes

*I generally use **Mouser.com** for my parts since they have a huge selection and really granular search options. It's not always the friendliest site for a hobbyist to order from, though.*

*NTE PartsDirect (**www.ntepartsdirect.com**) has a good selection as well, and their passive components like capacitors and such are marked by common voltages. Plus, it's pretty easy to navigate.*

There are a few other smaller suppliers catering to the hobbyist radio repair market. Just Radios

*(**www.justradios.com**) or Sal's Antique Radios (**www.tuberadios.com**) carry commonly needed sizes for vintage radio repair. Since they're very focused, you don't have to wade through a ton of options to find what you're looking for.*

*I generally purchase tubes from Antique Electronics Supply (**www.tubesandmore.com**) as they have new and good used tubes at pretty reasonable prices for most types.*

*If I need controls or switches rebuilt, I generally use Antique Audio/Mark Oppat's Old Radio Parts (**www.oldradioparts.net**). He's an expert in controls who can rebuild or custom build nearly any value of control.*

Continued on page 80

Make up to \$100 an Hour or More!



Be an FCC LICENSED ELECTRONIC TECHNICIAN!

Get your "FCC Commercial License" with our proven Home-Study Course!

- No costly school. No classes to attend.
- Learn at home. Low cost!
- No previous experience needed!
- **GUARANTEED PASS!** You get your FCC License or money refunded!



Move to the front of the employment line in Radio-TV, Communications, Avionics, Radar, Maritime and more.

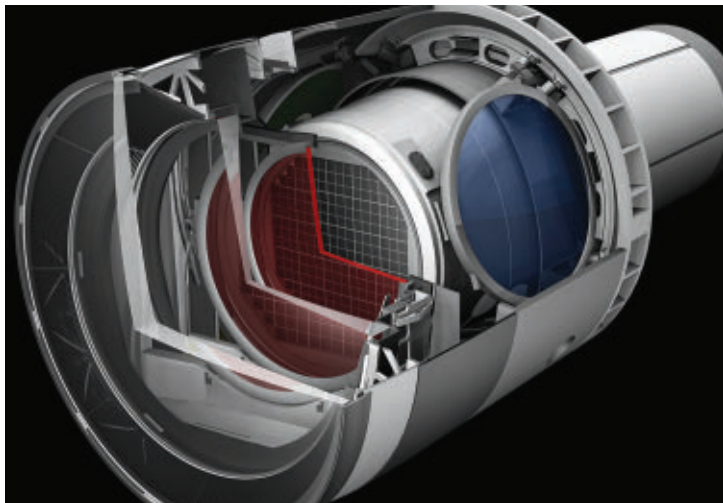
Call for FREE info kit: 800-877-8433 ext. 109

or, visit: **www.LicenseTraining.com**

COMMAND PRODUCTIONS • FCC License Training
Industrial Center, 480 Gate Five Rd., PO Box 3000, Sausalito, CA 94966-3000



ADVANCED TECHNOLOGY



■ Artist's rendering of the LSST camera.

World's Largest Digital Camera Funded

If you are in the market for a great pro-level digital camera, you might take a look at the Nikon D4: a 16.2 MP unit that lists for \$5,999.99 (plus lenses). It's pretty impressive, but how about a 3,200 MP camera — the final cost of which has been estimated at about \$167 million? We're talking about the centerpiece of the Large Synoptic Survey Telescope (LSST) which recently received key "Critical Decision 2" approval from the US Department of Energy. The camera — scheduled to be in operation atop a mountain in Cerro Pachón, Chile, by 2022 — will be the size of a small car and weigh more than three tons.

Operated by the DOE's Stanford Linear Accelerator Center (SLAC) and National Accelerator Laboratory (www6.slac.stanford.edu), it is designed to help researchers study galaxy formation, track hazardous asteroids, grab a look at exploding stars, and gain a better understanding of enigmatic dark matter and dark energy which together make up 95 percent of the universe. According to SLAC, it will produce the widest, deepest, and fastest views of the night sky ever observed. Over a ten year period, the observatory will detect tens of billions of objects and create movies of the sky with unprecedented details.

"The telescope is a key part of the long-term strategy to study dark energy and other scientific topics in the United States and elsewhere," said David MacFarlane, SLAC's director of particle physics and astrophysics. "SLAC places high priority on the successful development and construction of the LSST camera and is very pleased that the project has achieved this major approval milestone."

Components of the camera will be built by an international collaboration of labs and universities, including DOE's Brookhaven National Lab, Lawrence Livermore National Lab, and SLAC itself, where the camera will be assembled and tested.

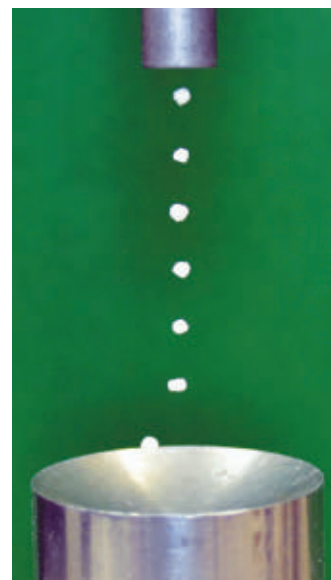
Levitation Made Simple

Levitation has long been a staple concept for folks, ranging from Hindu gurus to magicians. It is a pretty difficult thing to achieve outside the world of illusion and fakery, but there are a few real methods including magnetic and acoustic levitation — the latter of which is used for "containerless processing" of various materials. It involves suspending an object between a sound source and a reflector using the reflected acoustic waves.

Until recently, this has required a very precise setup in which the source and reflector had to be located at fixed resonant distances, which makes it difficult to control the levitated objects. However, researchers with the University of São Paulo (www5.usp.br) have devised an instrument that can make a small object hover while exercising much greater control over it.

The instrument demonstrates that it is possible to build a practical nonresonant device, i.e., one that does not require a fixed separation between the two main components. Moreover, it shows that levitation can do more than just trap objects in a fixed position; it can also transport them through short distances in space. According to Marco Aurélio Brizzotti Andrade, who led the research, "Modern factories have hundreds of robots to move parts from one place to another. Why not try to do the same without touching the parts to be transported?"

Unfortunately, his levitator can lift only very light objects at this point; it was tested with 3 mm blobs of polystyrene. "The next step is to improve the device to levitate heavier materials," Andrade noted. ▲



■ Ultrasonic emitter (top) and reflector (bottom) suspend polystyrene blobs in acoustic standing waves.

COMPUTERS and NETWORKING

Intel Gets Sticky

It may look like an Amazon Fire TV Stick or a similar streaming device, but the Compute Stick from Intel (www.intel.com) is actually a real live computer that comes with your choice of Windows 8.1 or Linux preinstalled. Powered by a quad-core Atom Z3735F processor, the Windows version also features built-in 802.11bgn Wi-Fi and Bluetooth 4.0, 2 GB of RAM, 32 GB of storage, a microSD slot, and a USB 2.0 port — all for \$149.

If even that doesn't fit your budget, you can opt for the Ubuntu Linux version which goes for \$89, but you only get 1 GB of RAM and 8 GB of storage.

Either way, just connect it to the HDMI and USB (for power) ports of your TV set, hook up a USB or Bluetooth keyboard, and you're ready to run your favorite applications. And, of course, you can also stream Netflix, Hulu, or games. ▲

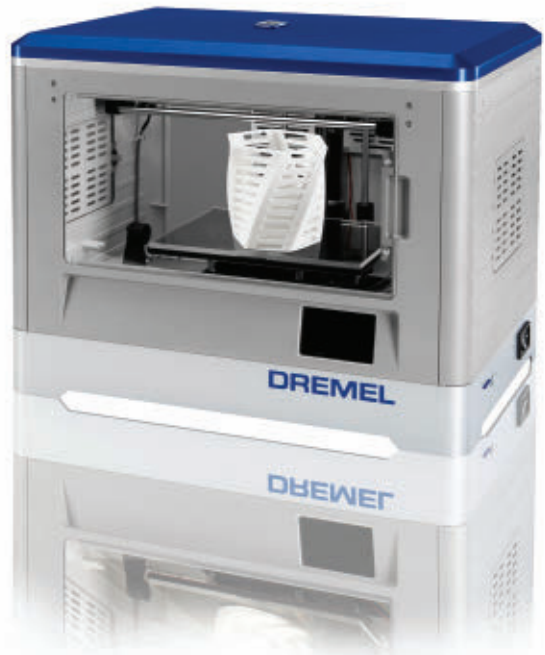
■ Intel's Compute Stick, the world's smallest Windows computer.



3D Printer with Autodesk Models

If you've been thinking about getting a 3D printer but have been reluctant to buy one from a company that operates out of a garage in Podunk, maybe the name Dremel (www.dremel.com) will instill some confidence. Billed as the most user-friendly unit on the market, the 3D Idea Builder is designed to inspire and empower the end-user to build on their own ideas with the support and mentorship of the Dremel experts. The company, in partnership with Autodesk, provides free print-ready models and design tools with the unit, and offers a flow of new design tools on dremel3d.com.

The machine features a color touch screen and onboard print software, plus, the work area (9 x 5.9 x 5.5 in) is fully enclosed to reduce noise. A cooling fan prevents warping of the workpiece, and the print head provides 100 micron resolution and a choice of 10 colors of PLA filament. You can pick one up at Home Depot or order one online, but it will set you back a fairly substantial \$999. ▲



■ Dremel's 3D Idea Builder printer.

CIRCUITS and DEVICES

USB 3.1 Type C Arrives

In case you haven't heard, USB connectors have been majorly redesigned. The new Type C connector showed up at the International Consumer Electronics Show back in January with pretty much universal acclaim. For one thing, both ends can be identical — unlike a micro USB — so you can plug either end into a compliant camera, computer, or whatever. Another nice feature is that it is reversible, so you won't end up trying to plug it in upside-down on the first attempt.

In addition, it is compatible with USB 3.1, which operates at up to 10 Gbps and can deliver up to 100W of power. The only problem, of course, is that you probably don't have anything that can use it yet. For details, visit www.usb.org. ▲



Skin Care via Your iPhone

Is your skin itchy, broken out, or infested with a rash that won't go away? Do you have painful boils or pustules that cause other people to make retching sounds? Well, you're in luck, because Oku (getoku.com) is here. Said to be based on a combination of dermoscopy,

nanotechnology, and spectroscopy (no details are offered as of this writing), Oku is an iPhone dongle devoted specifically to examining and maintaining the wellness of human skin. All you have to do is connect it, download the app, and let it do the rest.

According to the website, "Oku sees what you can't — by literally looking under your skin. It takes a scan of your skin, analyzes it in detail, takes into account your lifestyle information, and provides you with an easy to understand value called the SkinScore. This will tell you how your skin is faring and identifies areas for improvement. It then sets a daily goal towards unlocking the youthful best of your skin. Oku gives advice on your lifestyle and diet, and will recommend the right products for your current issue, or the right routine to improve your skin wellness." The bad news is that it will run you \$299 unless this gets to you before the \$249.95 preorder offer runs out. ▲



Wi-Fi Range Booster

If you're tired of losing your Wi-Fi signal every time you go out to the mailbox or down into the basement, the solution might be the new TAP-EX2 touch screen range extender (also referred to as the model AC750) from Amped Wireless (www.ampedwireless.com). Designed to work with any Wi-Fi network, it eliminates many dead spots and provides faster 802.11ac connections. Using six amplifiers and two high-gain antennas, the 800 mW unit is said to provide as much as 10,000 ft² of extended coverage. The touch screen is used to control all functions, including setup and management of guest networks, access scheduling, user access controls for Wi-Fi strength, and other parameters. Notably, the unit was chosen as a 2015 CES Innovations Design and Engineering Awards Honoree at the Las Vegas show. The EX2 was introduced at the show, but no MSRP was revealed. The previous model, however, was priced at about \$120. ▲

■ The new TAP-EX2 Wi-Fi extender from Amped Wireless.



CIRCUITS and DEVICES Continued

Outpeep the Peepers

Of course you are not involved in any illegal, immoral, or otherwise embarrassing activities that you don't want anyone to know about. Certainly not. But if you were, you might be alarmed to know that several companies manufacture optical devices that let police or other snoop people reverse the function of door peepholes and see (and even photograph) exactly what's going on inside.

However, for every tactic, there is a counter-tactic. So, if indeed there is something fishy going on (or you're just paranoid), check out the Brinno (www.brinno.com) PHV132512 Digital PeepHole Viewer.

Not only does it prevent visitors from spying on you, it turns the peephole image into a big bright digital image on the unit's display. It also compensates for low-light conditions and eliminates fisheye distortion.

To conserve battery life, the viewer automatically shuts down after 10 seconds. It installs easily in doors from 1.375 to 2.25 inches thick, and the two AA batteries are included. It lists for \$129.99, but scrounging around the Internet turned up prices as low as \$89.95. A small price to pay if you and Agnes (the goat) need a little privacy. ▲



■ The Brinno PeepHole Viewer ensures privacy.



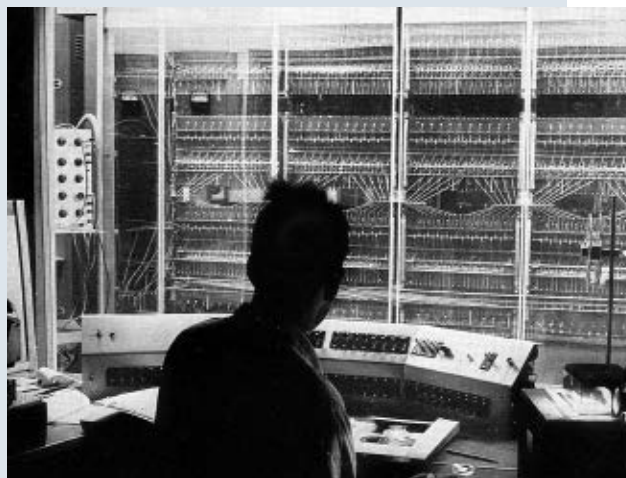
INDUSTRY and the PROFESSION

Computer Hits Age 60

Most of us are aware of the Electronic Numerical Integrator And Computer (ENIAC) which became the first electronic general-purpose computer in 1946. Fewer are familiar with the Weizmann Automatic Computer (WEIZAC) which debuted 60 years ago. Based on the Institute for Advanced Study (IAS) architecture created by John von Neumann, it was Israel's first computer and one of the world's first large-scale, stored-program machines in the world. The machine operated on 40-bit words and used punched tape for I/O (later, magnetic tape). Memory consisted of a magnetic drum that could hold 1,024 words (later expanded to 4,096).

The program to develop WEIZAC was initiated by Prof. Chaim L. Pekeris, who wanted to use it to solve Laplace's tidal equations for the Earth's oceans and other tasks, including defense-related ones. Eventually, it was also used for earthquake studies, atomic spectroscopy, numerical analysis, and so on.

Interestingly, although von Neumann supported the project, fellow Applied Mathematics Department committee member, Albert Einstein thought it was a bad idea. WEIZAC was fired up in 1955 and operated through 1963. In 2006, it was recognized by the IEEE as a milestone in electrical engineering, and the team who put it together was awarded the "WEIZAC Medal." NV



■ Israel's first computer, the WEIZAC.

In this column, Tim answers questions about all aspects of electronics, including computer hardware, software, circuits, electronic theory, troubleshooting, and anything else of interest to the hobbyist. Feel free to participate with your questions, comments, or suggestions. Send all questions and comments to: Q&A@nutsvolts.com.

- **Making a "Smart House" Even Smarter**
- **Telephone Off-Hook Alarm**
- **Radio Tuning**

Post comments on this article at www.nutsvolts.com/index.php?/magazine/article/april2015_QA.

Making a "Smart House" Even Smarter

Whenever I see a version of "smart house" advertised, I check to see if it has what I need, and it never does. I want something that will tell me when my microwave oven stops, when my dryer stops, and when my washer stops — without modifying the appliances themselves. I've made little boxes to plug microwaves and dryers into, and they let me know when these finish, but the washer eludes me.

Sometimes the motor is taking a lot of current, but then in the next second, all that's running is the timer to get to the next action. I know that a GFCI handles a lot of current while detecting a small amount of current; is there a way to change this circuit to tell me when the washer is completely finished?

— John Harris
Anaheim, CA

The automatic clothes washing machine was a real boon to laundry day because it did not have to be constantly attended (contrast with the wringer type machine, rub board and wash pot, or creek and rock methods). In the 1950s, my mother bought a wringer type washer and though it was fun to watch, it was a pain to use (literally at times, when she

caught her hand in the wringer). Modern washing machines perform the wash cycle without human intervention and some even weigh the contents of the tub to determine how much water, detergent, bleach, etc., to use. My washer has a signal that sounds at the end of the operating cycle. Most automatic washing machines have a sequence of cycles controlled by a timer that is roughly the same as what's shown in **Table 1**.

As you can see from the washing machine cycles, the current to the machine varies greatly throughout the operating sequence. So, using current measurements to determine when the cycle is finished is not a viable option (microwaves and dryers only turn ON and OFF once during a standard operation). My washer has a signal which is activated by a contact on the timer, but if you do not have a signal already, your timer most likely will not have this contact. (**Caution:** Unplug the washer before opening the control box. This is the easiest option, so check the timer schematic to be sure).

One idea I have — if you want to use a "box" that plugs between the receptacle and washer plug — is to use a current transformer around *one* wire going to the machine [see "Measuring Current with Clamp-On Ammeter"/Q&A October 2014 for more info]. If the signal is too small, use a pre-amp. Use a full-wave bridge rectifier to convert the AC to DC, and a Schmitt trigger to square the waveforms. Count the number of current (or voltage) cycles with a pre-settable digital counter and turn in a 555 one-shot to activate a piezo buzzer for a short time (my washer has eight or nine operations per cycle, depending on if you select extra rinse or not) when the counter pre-set is reached. Be sure to isolate the digital electronics from the 110V AC power. The GCFI only detects a mismatch in current between the hot and neutral leads such as when one of the leads is grounded and the other is not. In normal operation, the GCFI does not give any indication that would be useful in your indicator [see "Testing GCFIs Properly" Q&A January 2015 for more info].

Since most dryers and microwave ovens have built-in buzzers to indicate the cycle is finished, I think you may be looking for a visual indication so you could substitute an LED (with proper current-limiting resistance) for the piezo buzzer. An illuminated LED would let a hearing impaired person know when the washer was finished if

Start	Timer starts moving to initiate the cycles (timer energized — very low current)
Fill	Water solenoids open and fill tub to desired level (solenoid on — low current)
Wash	Agitator operates to slosh dirty clothes (motor on — high current)
Drain	Pump operates to remove water from tub (motor on — high current)
Fill	Water solenoids open and fill tub to desired level (solenoid on — low current)
Rinse	Agitator operates to slosh clothes to remove detergent (motor on - high current)
Spin	Tub spins to remove water from clothes to aid in drying (motor on — high current)
Off	Timer stops moving (timer de-energized — no current on most machines, I would think)

Table 1.

they were in the vicinity of the machine, of course.

In my case, the washer cycles in 30 minutes versus 60 for the dryer. So, once the first load is put into the dryer, I don't need to know when the washer cycle is over (most of the time). As one of my idiosyncrasies, from the den I can tell when the washer finishes because it makes a unique "clunk" only at the end of the cycle.

Your mention of a "smart house" reminds me of the magazine articles that spend many pages touting the marvels of a smart home, such as when you can have the coffee maker prepare your morning coffee while you sleep or turn the oven on to cook a roast for dinner. However, someone has to put the roast into the oven, or put coffee and water into the coffee maker.

My philosophy is that people are getting too lazy and need some kind of daily physical activity (besides a gym visit) to keep their bodies working as they should. I remember an EE professor in the early '70s talking about the "recent" developments in computer-aided circuit design. He said, "This may be an example of a technology that destroys the fundamentals that created it." Technology is great, but many things in our lives need a human touch.

Let me know how this idea works.

Telephone Off-Hook Alarm

Q Some of us who still have land lines for one reason or another (security system, etc.) sometimes leave a phone off-hook. How about a system to remind you that the off-hook situation exists? Since no audio is on the line (just off-hook voltage), the off-hook warning has to trigger on voltage only, no signal. A pause in conversation has to be considered, and the warning/alert needs to last until it's shut off by the occupant. The off-hook voltage varies between different carriers; the off-hook voltage for the privately owned carrier I have is approximately 13.0 volts. AT&T is approximately 8.6 volts.

— Robert
via e-mail

A Land line telephone systems operate with an on-hook voltage of approximately 48 VDC, an off-hook voltage of 3 to 9 VDC with a current draw of 15 to 20 milliamps, and a ringing voltage of 90 VRMS AC at 20 Hz (these values vary somewhat between phone companies and distance from the central office). When the phone is off-hook and idle for a length of time determined by the phone company, the phone system will send out an obnoxious tone and a message to alert the user that they have left the phone handset off the hook, thus rendering the phone useless.

In the North American Numbering Plan, a quad-frequency tone is used consisting of frequencies 1,400 Hz, 2,060 Hz, 2,450 Hz, and 2,600 Hz, with a duty cycle of

50 percent (0.1s on, 0.1s off). If I were to design an off-hook alarm device, I would use the phone company's alarm (the hard work has already been done, so let's use their signal) and I would use a high pass filter with a cutoff frequency of 200 Hz to eliminate the ringer signal, but pass the off-hook signal from the phone company (including those that still use the old GTE 480 Hz alarm). This filter would feed either an audio amp circuit (if you want an audio alert) or a full bridge rectifier/capacitor circuit feeding an LED for a visual circuit or a PIC detector (if you are into programming).

Fortunately, this circuit has already been incorporated into commercially-available devices such as the Serene Innovation HD-60 at \$100+. The HD-60 is pricy but it is ready to go out of the box, hearing impaired compatible, and approved for use with telephone systems. Most telephone providers are extremely adverse to customers installing unapproved devices on their systems without proper isolation (a.k.a., unapproved equipment). This sounds like a draconian measure, but it is designed to protect the technicians working on the line from electrocution (the power company has similar restrictions for customer devices connected to their power grid).

I found several cheaper indicators, but they all illuminate all of the time the phone is off-hook, so they do not meet your requirements to activate when the line has not been used for a period of time.

Let me know if this satisfies your needs.

Radio Tuning

Q I have a small transistor from which I can only receive static. I opened the radio up and noticed there are several square "cans" that have a slotted screw on the top. Will turning one of these screws allow the radio to receive AM stations again?

— Barbara Weathers
Kissimmee, FL

A I remember my first pocket-sized transistor radio from the early '60s. I was a teenager, and the tiny black radio only received the AM (Amplitude Modulated) band (for you youngsters, this is the band you don't use on the AM/FM radio) and had to be positioned properly to receive a signal well. This radio was great because it was portable since it was powered by a single nine volt battery (which I still call a transistor radio battery). The other radios of the day were desk or table models which plugged into the house power receptacle, so the transistor radio was a truly mobile device. I was reading everything I could about radio and television electronics at the time, so I naturally opened up the radio one day and admired all of the neat tiny electronic components.

Desktop radios were loaded with tubes/sockets, huge resistors/capacitors, and point-to-point wiring. They were

easy to fix by replacing a tube, but not too mobile unless you had a very long extension cord. Plus, they weighed more than the latter day boom boxes. (I'm not old like it seems. I just hav8e lots of experience.) Naturally, my attention was attracted to the square silver "cans" with the brightly colored slotted screws, but I soon learned to NEVER turn one of those screws if you ever wanted the radio to work again.

The AM radio detects radio frequency signals (RF) from 540 to 1,600 kHz by mixing this input signal from the antenna tuned circuit (sometimes one of the "cans") with a local oscillator circuit (another possibility for the cans) to produce an intermediate frequency signal (IF) at 455 kHz which was more efficient to amplify the RF signal. The other two cans were used to tune the two stages of circuits that fed the IF into the rest of the radio.

The tuning is essential to creating a narrow band filter at the correct resonant frequency which eliminates extraneous signals created by the heterodyning process or spurious RF channels. Interestingly enough, the demodulator for the AM signal could be a diode (FM demodulation is a LOT more involved). Tuned is the key word to why you NEVER touch the screws on these cans (really they are variable transformers with movable ferrite

slugs). The RF and IF transformers (and sometimes the local oscillator) are used to adjust the coupling between the primary and secondary windings and thus the width of the passband (greater coupling gives wider and flatter passband; the capacitor sets the center frequency of the filter passband). So, if you ham-fistedly turn a screw, you now are operating with a non-optimum passband. You will either not hear the radio station's broadcast (this is the voice of experience) or there will be excessive noise.

To correct this problem, you need a circuit schematic (to find the correct test points), an RF/IF signal generator to input a known signal (I happen to have some '60s vintage military surplus generators), and a voltmeter, speaker, or oscilloscope to verify when the maximum output is reached from the IF and RF/LO circuits. The high end receivers have a manual that specifies the passband needed for best operation.

Most transistor radio problems are related to the transistors, diodes, resistors, capacitors, fixed inductors (coils and transformers), or the circuit board. I once fixed a squealing problem by touching each resistor with a pencil eraser until the problem stopped – revealing a cracked resistor which could not be seen with my (then) youthful naked eye. I hope this answers your question. **NV**

BEST SCOPES, BEST PRICES



PASSPORT-SIZE PC SCOPES \$149+
Great scopes for field use with laptops. Up to 200MHz bandwidth with 1GSa/s, high speed data streaming to 1MSa/s, built-in 1GSa/s AWG/wfm gen. **PS2200A series**



30MHz SCOPE \$289
Remarkable 30MHz, 2-ch, 250MS/s sample rate scope. 8-in color TFT-LCD and AutoScale function. Includes FREE carry case and 3 year warranty! **SDS5032E**



100MHz SCOPE \$830
High-end 100MHz 4-ch 1GSa/s oscilloscope with 12Mpts memory, USB port, 7" display and Rigol's UltraVision technology. Includes FREE carry case! **DS1104Z**

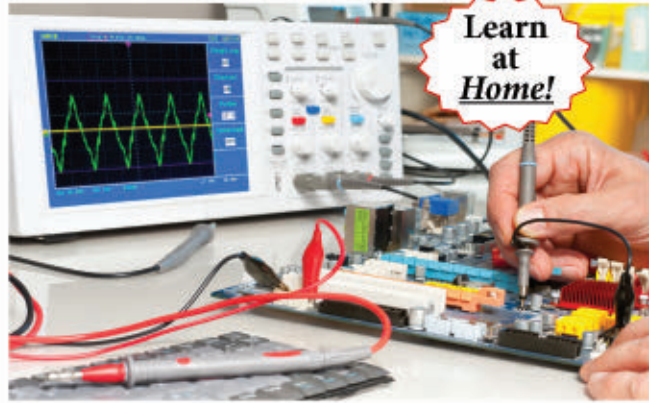


VERSATILE PC SCOPES \$525+
2/4-ch 50-200MHz oscilloscopes with built in function/arbitrary waveform gen and MSO options that combine fast sampling and huge buffer memory to 512Msa. **PS3000D series**



70-300MHz SCOPES \$839+
Fast, versatile 2-ch 2GSa/s scopes with 8-in WVGA LCD, integrated generator, 14Mpt memory, very low noise floor. FREE carry case available! **DS2000A series**

FREE TECH SUPPORT - GREAT CUSTOMER SERVICE 



Learn at Home!

Electronics Courses

Cleveland Institute of Electronics

Train at home to be a professional electronics or computer technician! Learn with hands-on labs, instructor support, online exams, videos and instructor led chat rooms.


FREE course catalog www.cie-wc.edu or (800) 243-6446

- **NEW!** Robotics Automation Lab
- Industrial Electronics with PLC
- Electronics with FCC
- Electronics Troubleshooting

- **NEW!** Computer Security Specialist
- Broadcast Engineering
- PC Troubleshooting
- Networking

Visit www.cie-wc.edu and start today!

CIE: 1776 E. 17th St., Cleveland, OH 44114 • (800) 243-6446 • Registration 70-11-0002H



WORLD'S MOST VERSATILE CIRCUIT BOARD HOLDERS

Model 324 Our Circuit Board Holders add versatility & precision to your DIY electronics project. Solder, assemble & organize with ease.

VISIT US ON



MONTHLY CONTEST
 Visit us on Facebook® to post a photo of your creative PanaVise project for a chance to win a PanaVise prize package.

PANAVISE®
 Innovative Holding Solutions

7540 Colbert Drive • Reno • Nevada 89511 | (800) 759-7535 | www.PanaVise.com




2 Layers

\$3

ea.

- FULL DRC CHECK
- CAD/CAM REVIEW
- SOLDER MASK ON BOTH SIDES

Accutrace inc.

www.PCB4u.com
 (408) 748-9600

PRINTED CIRCUIT BOARDS



SUPERIOR EMBEDDED SOLUTIONS



DESIGN YOUR SOLUTION TODAY
CALL 480-837-5200

www.embeddedARM.com

TS-7670 and TS-7680 Industrial Computers

- Up to 454 MHz ARM w/ 256 MB RAM
- 2 GB Flash Storage
- Industrial Temperature (-40 °C to 85 °C)
- DIO, CAN, Modbus, RS-485

TS-7670 Features:

- GPS and Cell Modem
- 1x Ethernet
- 2x microSD Card Sockets

Pricing starts at

\$129
Qty 100

\$168
Qty 1

NEW!

TS-7680 Features:

- WiFi and Bluetooth
- 24 VAC Power Input
- 2x Ethernet

Low cost plastic enclosure available



NEW! TS-4900 High Performance Computer Module

- Up to 1 GHz Quad Core ARM CPU
- Up to 2 GB DDR3 RAM
- WiFi and Bluetooth
- 4 GB Flash and microSD
- Gigabit Ethernet
- SATA II and PCI-Express
- DIO, CAN, COM, I2C, I2S
- LVDS & RGB Display Ports
- Industrial Temperature (-40 °C to 85 °C)
- Supports Linux, Android, & QNX
- Windows Support Coming Soon

Pricing starts at

\$99
Qty 100

\$134
Qty 1



NEW!

TS-7970
SBC Version of the TS-4900



We've never discontinued a product in 30 years



Embedded systems that are built to endure



Support every step of the way with open source vision



Unique embedded solutions add value for our customers

Can We Talk? Again?

I've concluded — like many people my age — that as we grow older and wiser, we become more honest and objective with ourselves, often being forced to admit that our "babies" are not as cute and bright as we once thought. I'm not talking about our human babies, of course; my own son is a very handsome and talented young man. What I'm talking about is our software babies. A couple years ago, I created what I proudly proclaimed as the Human Friendly Control Protocol (HFCP). When my friend, Lon called to ask for some help, I hadn't used HFCP myself in a while. On review, I had one of those "What was I thinking?" moments.

That's not to say that HFCP is bad. It isn't — it's just not as friendly as my memory led me to believe. A friend of Lon's actually deployed HFCP but has since moved on, leaving Lon and me to sort things out. In addition to this, another project we're working on together requires configuration files on an SD card attached to the Propeller. HFCP is designed for command and control, not processing text from files. I needed a new tool.

What I created is a generic parsing engine; one that I could use on-the-fly for command and control, and one that I could pass complete strings to that might be built into a program or are read from a file (that was actually very easy). Like HFCP, everything is text based. We can use a simple terminal to communicate commands to a project, and the responses can be as verbose as we like. Again, it's plain text. For projects with an SD card attached, we can open and process text files.

Every coin has two sides, and the other side here is that the easier we want to make the interactions for humans, the more work we have to do in the code. This month's project goes through setting up a simple command language (we're going to use PBASIC-like commands) for controlling the Propeller through a serial interface.

Ready ... Set ... Action!

Let's jump right in, shall we? In a live application, we're going to accept a character from a serial stream and pass it to the `parser.enqueue()` method. This method will store the incoming character in a string buffer; when a line terminator (CR, LF, or 0) is detected, the string will be parsed. When this happens and tokens are available, the `parser.enqueue()` method returns `true`.

In an application, we'll usually see something like this:

```
repeat
  c := serial.rx
  if (parser.enqueue(c) == true)
    process_cmd
    parser.reset
```

As you can see, we're waiting on a character from a serial stream. When that arrives, it gets passed to the parser. When tokens are ready (result is `true`), we call `process_cmd` and then reset the parser for the next command. Mind you, the serial data can come from anywhere; our demo code will use the programming port, but it could just as easily be coming from an \$8 Bluetooth device running SPP mode. I've been doing that and it's a lot of fun.

For clarity, when I use the term 'token' I am referring to a group of valid characters. Any other characters are not valid; hence, used as separators. This allows us, for example, to parse values out of a CSV (comma-delimited text) file that might use a comma and space between fields. The point is we don't have to define special separator characters; anything not valid is taken to be a field separator. This string:

HIGH 26<CR>

has two tokens: 'HIGH' and '26.' The space character is not part of the valid set, so is used as a separator.

How does the parser know what's valid and what's not? We tell it, of course, using its `parser.start()` method which looks like this when implemented:

```
parser.start(@CHAR_SET, {
} @TOKEN_LIST, {
} TOKEN_COUNT, {
} false)
```

The `parser.start()` method takes four parameters: 1) a pointer to a string that defines the valid character set for the application; 2) a pointer to a list of tokens used by the

Post comments on this article and find any associated files and/or downloads at www.nutsvolts.com/index.php?/magazine/article/april2015_SpinZone.

BOM

ITEM	DESC.	SOURCE/PART #
Propeller Activity Board		Parallax #32910
Bluetooth (RN-42)		Parallax #30086
Bluetooth (RN-42, Xbee style)		SparkFun #WRL-11601
Bluetooth (HC-06)		Amazon.com or ebay.com

application; 3) the number of tokens the application can use; and 4) a true or false value that sets case sensitivity. In most applications, we will use **false** which causes the parser to convert everything to uppercase — this simplifies command processing.

The valid characters and token strings are defined in a **DAT** table:

dat		
CHAR_SET	byte	"\$%+-_"
	byte	"0123456789"
	byte	"ABCDEFGHIJKLM"
	byte	"NOPQRSTUVWXYZ"
	byte	0
TOKEN_LIST		
TOKEN_HIGH	byte	"HIGH", 0
TOKEN_LOW	byte	"LOW", 0
TOKEN_ON	byte	"ON", 0
TOKEN_OFF	byte	"OFF", 0
TOKEN_FREQ	byte	"FREQUOT", 0
TOKEN_BLINK	byte	"BLINK", 0

The character set is a contiguous string, even if broken across multiple lines. The only place we find the 0 terminator is at the very end. The list of tokens on the other end is a set of individual strings. I add in a label called **TOKEN_LIST** so that I can rearrange tokens as I choose without having to update the call to the **parser.start()** method.

In practice, I tend to order the list by expected use; that is, tokens that are used frequently are at the top of the list. This reduces search time for popular tokens.

You may be wondering about words versus numbers. Yes, the character set we're using allows both words and numbers.

In fact, beyond numbers and letters, the only other valid characters we have in this program support Parallax-style numeric formatting. The parser will evaluate 26, \$1A, and %1_1010 the same way.

We can use **parser.token_is_number()** to determine if a token is a number, and if it is, use **parser.token_value()** to convert it from text to a number. More on this later.

Under the Hood

Okay, let's have a look on the inside to see how this works. The front end of the parser is the **parser.enqueue()** method that accepts characters until an end-of-string character is detected.

As I want to be able to use this for live control from a terminal, it supports backspace (8) as well:

```
pub enqueue(c)

case c
  0, 10, 13:
    queue[qidx] := 0
    ifnot (csensitive)
      str.ucstr(@queue)
    tcount := count_tokens(@queue)
    if (tcount > 0)
      extract_all
      return true
    else
      reset

  8:
    if (qidx > 0)
      queue[--qidx] := 0

other:
  queue[qidx++] := c
  if (qidx == BUF_SIZE)
    qidx -= 1

return false
```

The **parser.enqueue()** method builds a string that is stored in a byte array called *queue*. The position index of the next available character is saved in *qidx*. When the new character is any of the terminators, it is changed to 0; the string is converted to uppercase if the *csensitive* flag is **false**; and the possible tokens in the string are counted. If the token count is greater than zero, they get extracted and we return **true** to the caller. If we had a bad string and there are no tokens, the parser resets itself and returns **false**.

For live control, I thought the input should be human friendly (*there's that term again!*) and support the backspace character so mistakes can be dealt with. If backspace (8) is entered and the buffer is holding characters (*qidx* is greater than zero), we back up the index and clear the last entry.

With any other character, we add it to the buffer — so long as there is space left. I've made the buffer size about twice as long as I think I need; this should prevent

Jon "JonnyMac" McPhalen
jon@jonmcphalen.com

Parallax, Inc.
 Propeller boards, chips,
 and programming tools
www.parallax.com

bumping into the end.

To this point, we've accepted any character into the raw string. Counting tokens requires the use of the valid character set and it's pretty easy. We're going to iterate through the buffer; when we go from an invalid (separator) character to one in the valid set, the token count is incremented:

```
pub count_tokens(p_str) | toks, tflag, c

  toks := 0
  tflag := false

  repeat strsize(p_str)
    c := byte[p_str++]
    if (str.cinstr(p_vchars, c) => 0)
      if (tflag == false)
        tflag := true
        ++toks
      else
        tflag := false

  return toks <# N_TOKENS
```

As you can see, it's a short method, taking advantage of the **str.cinstr()** method from the strings object. The return value is limited to the storage space for tokens; the default setting is 10.

Let's say we have some tokens — time to parse them from the string. At the heart of the process is **parser.extract_token()** which looks a bit gnarly, but is really not too bad:

```
pub extract_token(p_str, tid) | {
  } target, tflag, c, p_tok, tlen

  if (tid >= N_TOKENS)
    return
  else
    target := tid

  tflag := false

  repeat strsize(p_str)
```

```
  c := byte[p_str++]
  if (str.cinstr(p_vchars, c) < 0)
    tflag := false
  else
    if (tflag == false)
      tflag := true
      if (target > 0)
        --target
      else
        p_tok := token_addr(tid)
        bytefill(p_tok, 0, TOK_SIZE)
        byte[p_tok++] := c
        tlen := 1
      repeat
        c := byte[p_str++]
        if (str.cinstr(p_vchars, c) => 0)
          if (++tlen < TOK_SIZE)
            byte[p_tok++] := c
          else
            return
        else
          return
```

Again, the process is simpler than the code looks. What we're going to do is iterate through the buffer until we locate the target token (specified in *tid*). Once that's located, the storage space for that token (provided by **parser.token_addr()**) is cleared, and the characters in the string which make up the token are copied to the token buffer. As expected, any non-valid character ends the process.

It's logical to wonder why I didn't merge counting and parsing tokens into a single method. I may, in fact, do that later, but at the moment I want the flexibility to skip parsing tokens if the count does not match what an application is expecting (currently, parsing is automatic on detection of a string terminator).

By Your Command

Okay, that's enough of the black box stuff. Let's put this dude to use. The demo program allows us to give a few PBASIC-like commands to the Propeller. The keywords understood by the program are:

HIGH
 LOW
 ON
 OFF
 FREQOUT
 BLINK

This is where it gets fun: We get to decide the syntax rules for our commands. We've added ON and OFF which work the same as HIGH and LOW, and we're going to

supplement the syntax rules so that we can use them like PBASIC:

HIGH 26

Or, as "regular people" might prefer:

26 OFF

To be honest, I probably wouldn't have considered the second syntax variation, but something interesting happened while in a friend's shop a few weeks ago. He's working on a scoreboard for a baseball park and asked me about controls. During our conversation he very casually stated, *"I'd love to tell it something like, '3 balls' and have the scoreboard update."* This was a great piece of information from a possible user.

I stated earlier that the easier we make things on the outside (the human interface), the more work we have to do on the inside. Believe me, it's worth the effort.

In the parser demo, a method called **process_cmd** is called when **parser.enqueue()** or **parser.enqueue_str()** return **true**:

```
pub process_cmd | tidx

if (parser.token_is_number(0))
  process_pin
  return

tidx := parser.get_token_id(parser.token_addr(0))

case tidx
  T_HIGH, T_ON : set_pin(true)
  T_LOW, T_OFF : set_pin(false)
  T_FREQ      : pin_freq
  T_BLINK      : blink_pin
```

As you can see, there's not very much to it; this method is taking the first token and directing us to the appropriate handler for it. At the very top, we check to see if the first token is a [pin] number; if that's the case, we call the **process_pin()** method and then return to the main loop.

If the first token is a word, we use the **parser.get_token_id()** method to convert the token string into a numeric index that can be used in a **case** statement. To simplify the **case** structure, I create an enumerated constants list like this:

```
con

#0, T_HIGH, T_LOW, T_ON, T_OFF, T_FREQ,
T_BLINK
```

```
TOKEN_COUNT = T_BLINK+1
```

For the moment, let's start with processing a very simple command like **HIGH 26** which will turn on the LED on P26 of the Propeller Activity board. The index value to the "HIGH" token is 0; hence, the **case** statement will call a handler method called **set_pin()**:

```
pub set_pin(state) | pin

if (parser.token_count <> 2)
  return

if (parser.token_is_number(1))
  pin := parser.token_value(1)
  ifnot ((pin => 0) and (pin =< 27))
    return
else
  return

outa[pin] := state
dira[pin] := 1
```

In all of my handlers, the first thing I do is verify that the token count for the command is correct. For **HIGH**, **LOW**, **ON**, and **OFF**, we should have two tokens, and the second token must be a number. The pin number is extracted using the **parser.token_value()** method, and qualified for the available pins in the application. I only allow P0 through P27. This protects the I²C and programming/debug pins. If everything checks out, we write **state** to the pin and make it an output.

Want to have some fun? Update the **set_pin()** method to work with two or three tokens. When three tokens are used, the second and third tokens specify the boundaries of a group of pins to make high or low.

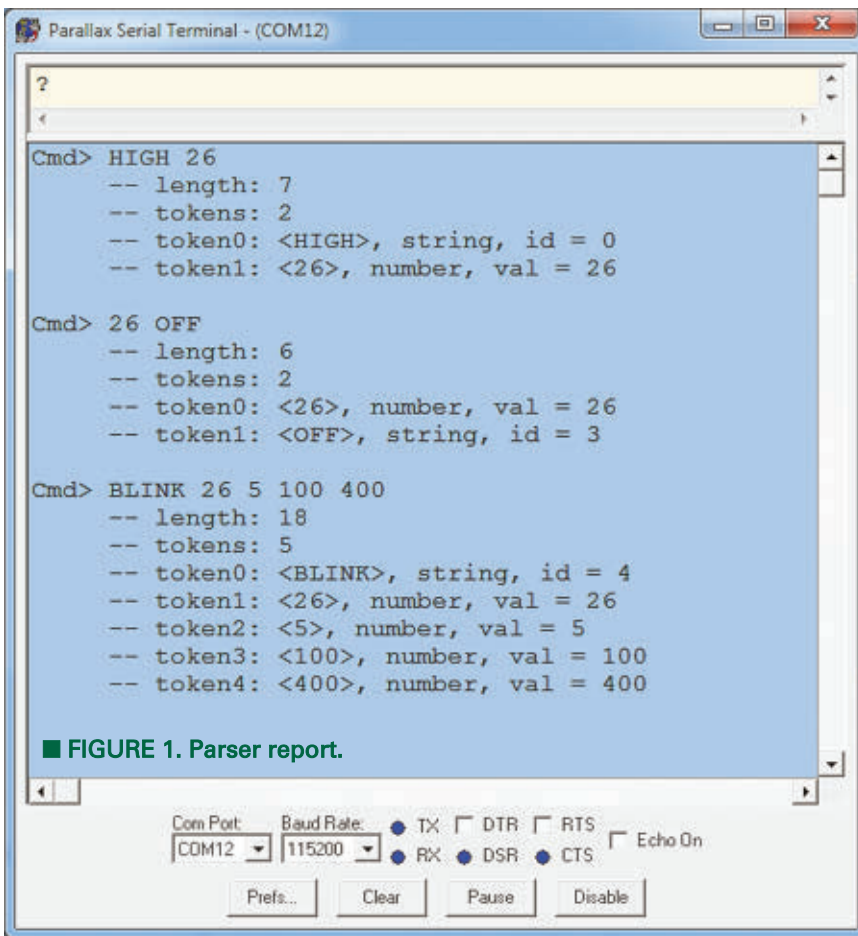
Here's one approach: If the token count for **set_pin()** is not two, call another method before returning to the main loop. I called that method **set_pins()**:

```
pub set_pins(state) | tc, lsb, msb

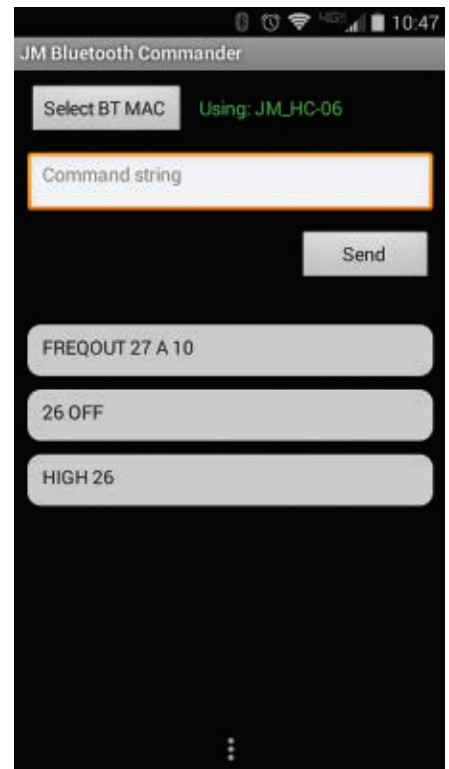
if (parser.token_count <> 3)
  return

if (parser.token_is_number(1))
  msb := parser.token_value(1)
  ifnot ((msb => 0) and (msb =< 27))
    return
else
  return

if (parser.token_is_number(2))
  lsb := parser.token_value(2)
  ifnot ((lsb => 0) and (lsb =< 27))
```



■ FIGURE 1. Parser report.



■ FIGURE 2. Android command app.

```

    return
else
    return

outa[msb..lsb] := state
dira[msb..lsb] := true

```

The structure is identical to the earlier method. In this case, we're expecting three tokens; the second and third must be numbers that are valid I/O pins for the application. Remember that **state** is being passed as **false** (0) or **true** (-1), and the internal constant **true** has all bits set so it will work with any size group of I/O pins. We use this to set all of the pins in the group to outputs via the **dira** register. Let's do one more that is a bit more involved. The **FREQOUT** command – as expected by the program – might look like this:

FREQOUT 26 A 10

The command takes four tokens. The second token is the pin number to use. The third token is the counter to use; this can be "A" or "B." The last token is the frequency in Hz. The above command would cause the LED on P26

to blink at a 10 Hz rate until we stop it by setting the frequency to 0.

Remember that counters work independently and in parallel to the normal pin output bits, so setting a pin to **LOW** or **OFF** will not stop an active counter attached to it.

Here's the code that processes the **FREQOUT** command:

```

pub pin_freq | pin, ctrx, hz

if (parser.token_count <> 4)
    return

if (parser.token_is_number(1))
    pin := parser.token_value(1)
    ifnot ((pin => 0) and (pin <= 27))
        return
else
    return

if (parser.token_len(2) == 1)
    parser.ucstr(parser.token_addr(2))
    ctrx := byte[parser.token_addr(2)]
    if ((ctrx <> "A") and (ctrx <> "B"))
        return
else
    return

```



```

if (parser.token_is_number(3))
  hz := parser.token_value(3)
else
  return

set_freq(pin, ctrx, hz)

```

Most of this should look familiar by now. The difference is in handling the second parameter (third token) which is the counter module to use. The command calls for a single letter, A or B. We start by checking the length of the token. If it is only one character long, we convert it to uppercase to simplify testing. Remember that tokens are stored as strings, so we use `byte[]` to extract the letter token from the string; the parser object includes a method called `parser.token_addr()` which returns the hub address where that token is stored. This allows us direct access to it.

Okay, it's your turn. Create a new command, add it to the token list, create the handler code, and give it a go. To aid the development of new commands, I have a reporting method that analyzes a new command and provides a breakdown of each of the tokens. **Figure 1** shows the report output from a few of the commands.

Easy Remote Control

As I stated earlier, the parser is generally taking input from a serial stream; it doesn't care where that serial stream is coming from. During development and testing, I used PST. I have a couple personal projects that would benefit from remote control — and this turned out to be a breeze. For remote input, I used a Bluetooth module. At the moment, I've tested three different units: the RN-42 module from Parallax; an XBee socket compatible RN-42 from SparkFun; and an \$8 HC-06 module that is available all over the Internet.

The control freak in me wasn't happy to use an off-the-shelf Bluetooth terminal on my phone, so I knocked up a little app using the MIT App Inventor 2. Sadly, this only works for Android phones. AI2 uses block-style programming which is not really my cup of tea, so I'm looking at other tools. Hopefully, I can find one that lets me deploy simple control apps on Android and iOS products. We'll see.

The \$8 HC-06 modules are very enticing for their price, so I bought one to try. Here's the rub: They're a bear to configure. When in command mode, it does not use a

terminating character (e.g., carriage return) for the command; it uses a serial timeout. What this means is that one cannot simply type commands in through a terminal.

I solved the problem — which also allows me to set up the HC-06 while in the project — by creating a configuration program that uses the parser engine. Changing the baud rate is now simple:

BAUD 38400

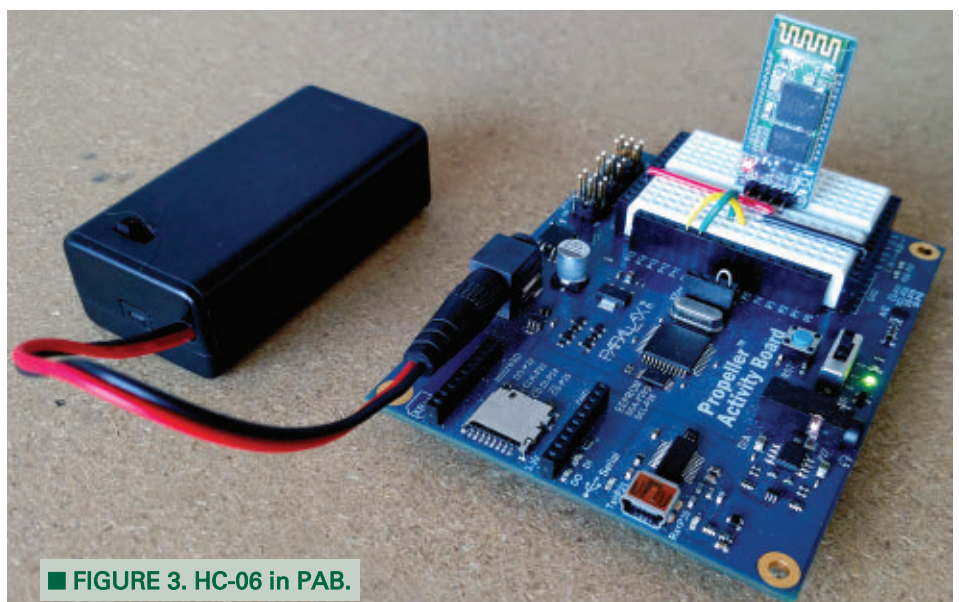
Other commands allow me to set the pairing code and the name — something I like to do with my projects. **Figure 2** is a screenshot from my phone showing that I'm paired with the HC-06 on my Activity board (**Figure 3**). I've included my RN-42 and HC-06 configuration programs in the downloads available at the article link.

On a Personal Note

I recently received happy and distressing news on the same topic on the same day. I learned that a friend's wife had successful surgery to remove a cancerous tumor and her doctors expect a full recovery — wonderful news. The same day I learned that another friend's wife has been diagnosed with terminal cancer — horrible news. Both of these ladies are young and have a lot to contribute to society.

Not many things truly upset me, but that people are still being taken by this horrible disease does. Please consider donating to cancer research; there are many organizations that can do great things with your generous donation. (Thank you for indulging my personal outburst.)

Until next time, keep spinning and winning with the Propeller! **NV**



■ **FIGURE 3. HC-06 in PAB.**

NEW PRODUCTS

■ HARDWARE
■ SOFTWARE
■ GADGETS
■ TOOLS

4WD OFF-ROAD ROBOT CHASSIS

ServoCity is now offering the Nomad 4WD off-road chassis kit which is an easy to assemble robotic platform capable of going places a normal chassis can't.

What makes this chassis kit different is not only its vast amount of attachment points for various add-ons, but also the fact that it is easily and fully configurable.

The chassis is equipped with four 5.4" diameter by 2.25" wide heavy duty tires, four super duty/ball-bearing planetary gearmotors with full metal gears, and a ball-bearing pivot suspension. Also included in the 4WD chassis kit is a large central ABS plastic body with two large access panels that open up to a cavity large enough for a 7.2V NiCAD/NiMH or other LiPo battery and electronics to fit comfortably. Its central body also has a multitude of 0.77" hub patterns and cutouts at the bottom to run motor wires through. Since it is made out of ABS plastic, drilling holes for additional mounting options is simple. Retail price is \$279.99.



arm. A force sensor (dynamometer) is used for controlling the force. This arm simplifies many mechanical devices where it's necessary to control the action force created by a servo drive or the action force of a mechanism controlled by a servo drive. When used instead of a servo drive with a controlled position of the arm, the arm makes it possible to improve the technical characteristics of many devices and mechanisms.

The key areas of application are remotely controlled light drones, robots, radio-controlled models, and rotation devices for video cameras. Retail price is \$24.99 each.

For more information, contact:

ServoCity

Web: www.servocity.com

FORCE SERVO ARM

Also available from ServoCity is the Force servo, which is a new type of a servo drive for remotely controlled devices. The majority of modern servo drives maintain a controlled position of the arm. Other servo drives are responsible for cyclic rotation with variable speed. Force Servo or F-Servo is a servo drive with controllable action force.

The value of force is proportionate to the control signal and does not depend on the position of the



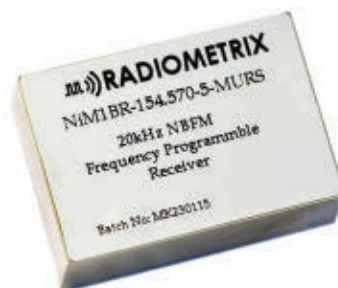
WIRELESS TRANSCEIVER

Lemos/Radiometrix now has available the new Mini-MURS (Multi-Use Radio Service) NiM1B-154.570-5-12.5-MURS which is a frequency programmable narrow band transceiver that offers a low power reliable data link in a Lemos/Radiometrix transceiver standard pinout and footprint. It is suitable for licensed and unlicensed VHF

allocations and Federal Communications Commission (FCC) part 90 and part 95.

Features include:

- Conforms to EN 300 220-3 and EN 301 489-3 (10 mW version only).



- Standard frequency 154.570 MHz or 154.600 MHz (re-programmable).
- Other frequencies from 120 MHz to 175 MHz.
- Data rates up to 5 kbps for standard module.
- Usable range over 1 km.
- Fully screened.
- Low power requirements.
- 25 kHz channel spacing.
- Feature-rich interface (true analog and/or digital baseband).

Applications include:

- Multi-use radio service.
- Industrial telemetry and telecommand.
- High-end security systems.
- Vehicle data up/download.
- ROV/machinery controls.

A technical summary is:

- Fully integrated sigma-delta PLL synthesizer based design.
- High stability TCXO reference.
- Transmit power: +13 dBm (20 mW)
- Image rejection: >70 dB
- Receiver sensitivity: -120 dBm (for 12 dB SINAD)
- RSSI output with >50 dBm range
- Supply: 3.3V - 15V @ 30 mA transmit, 18 mA receive
- Dimensions: 33 x 23 x 11 mm (fully screened)

For more information, contact:
Lemos International Co., Inc.
 Web: www.lemosint.com

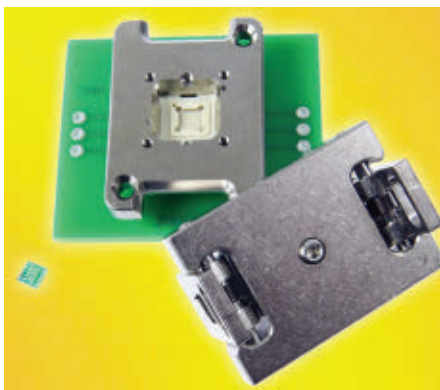
CLAMSHELL SPRING PIN QFN SOCKET

Ironwood Electronics recently introduced a new QFN socket addressing high performance requirements for testing QFN devices — the CBT-QFN-7039. The contactor is a stamped spring pin with 17 gram actuation force per ball, and a cycle life of 50,000 insertions. The self inductance of the contactor is 0.75 nH, insertion loss <1 dB at 31.7 GHz.

The current capacity of each

Recycling & Remarketing High Technology
WEIRDSTUFF®
 WAREHOUSE
 Software, Computers, Electronics, Equipment, Doohickies
 WE BUY/SELL EXCESS & OBSOLETE INVENTORIES
 FREE COMPUTER AND ELECTRONIC RECYCLING
 GIANT 10,000 SQ. FT. AS-IS SECTION
 384 W. Caribbean Dr.
 Sunnyvale, CA 94089
 Mon-Sat: 9:30-6:00 Sun: 11:00-5:00
 (408)743-5650 Store x324
WWW.WEIRDSTUFF.COM

In the Nuts & Volts Webstore NOW!
NUTS AND VOLTS
 NV BOOK SPECIALS
 Programming the Raspberry Pi
 Getting Started with Python
 Simon Monk
 Raspberry Three Book Combo
 Raspberry Pi User Guide
 Eben Upton
 Rachel Hoffman
 Raspberry Pi Projects for the EVIL GENIUS
 David Hux
Only \$48.95
 Plus
 FREE Priority Mail Shipping
 US Only
 To order call 800 783-4624 or visit:
<http://store.nutsvolts.com>
 Limited time offer



Continued on page 47

www.boxedkitamps.com
 100% start-to-finish amp kits,
 the highest quality parts,
 and anyone can build them.
boxed
 RELAXATION

NKC electronics
MDE8051 Trainer
 by Digilent
NKCElectronics.com/MDE8051
 Includes the MDE8051 training board, power supply, serial cable
 Purchase Orders are accepted from Educational Institutions, US Government and Research Centers

SDP Stock Drive Products
 Setting Ideas Into Motion
 One-Stop Shop for
 Mechatronic Components
 EXPLORE
 DESIGN
 BUY ONLINE
www.sdp-si.com
 no minimum requirement
Designatronics inc.

NATIONAL RF, INC.
 TYPE **RNF** 75-NS-3
MINI HF RADIO
 The 75-NS-3 covers 3.5 to nearly 11 MHz and is offered as a semi-kit.
CUSTOMER PUTS IT IN A MEAT CAN!!
 Visit www.NationalRF.com for this and other Radio Products!
 Office: 858-565-1319

BUILD YOUR OWN ARDUINO BAROGRAPH

In 1643, Evangelista Torricelli discovered that "we live at the bottom of a sea of air," and the pressure of the air could be measured with a simple column of water or mercury. In 1648, Blaise Pascal found differences in pressure with only slight elevation changes, leading to the idea that air has weight.

Changes in this pressure acting on evacuated bellows can be made to trace a line on a clockwork-powered drum, indicating fluctuations in atmospheric pressure as shown in **Photo 1**. You can still buy these devices which are credited to Lucien Vidi, who (in 1844) invented the practical barograph. They have been much improved, but still require a steady platform, winding of the clockwork, refreshing of the special ink (that must remain fluid until it hits the paper, then quickly dries), and renewing of the chart paper. They are also expensive.



Vidi is said to have spent years and all his money on the invention of the barograph. That was a time when the value of something was judged to be proportional to the time and effort that went into it. Of course, we in the modern era have been taught to laugh at ideas like that, and think ourselves superior because we can put together a better device with some parts bought online and a few hours' work.

I, for one, feel a quite satisfactory sense of accomplishment at having duplicated – if in only a removed sense – the device of Vidi with a graphical LCD, a BMP085 digital pressure sensor, and an Arduino Uno.

Photo 2 shows the breadboarded version. A resistor, a 10K pot, the GLCD, and the Arduino – that's it. The display lists the pressure in Pascals (named, of course, for the aforementioned pioneer), meters of elevation, and standard atmosphere fraction of the sea level average. Also, there are low and high graph limits.

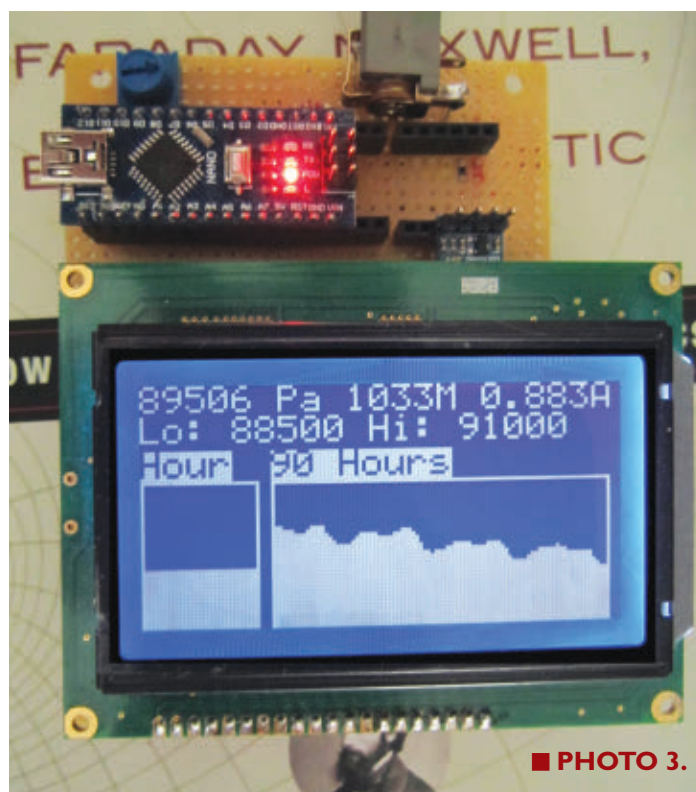
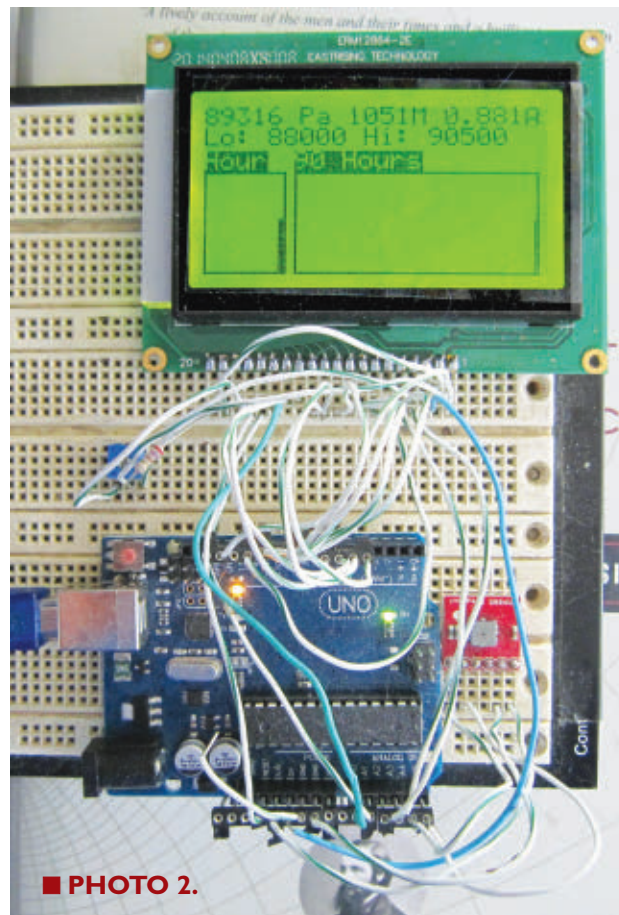
These limits are calculated like this: The *setup* function does a reading and then it rounds this up or down to the nearest 500 Pascals. It then adds 1,000 Pascals to the upper value, and subtracts 1,000 from the lower one. This is so the readings will have a chance of landing in the middle of the graph since the device may be at any altitude when it is started.

If the pressure happens to wander to within 100 Pascals of either limit, it adjusts both limits up or down by 500 Pascals. This creates an unavoidable discontinuity, but the jump is obvious on the graph. Refer to `90_hour_bmp085_GLCD_graph_baro.ino` at the article link.

There are two graphs: one for the last hour, to show rapidly changing pressure; and one for the last 90 hours, to show the long term trend. The graphing proceeds right to left, with the display marching as it were to the left as new values are added to the right. The program uses two circular buffers to store the readings, with the newest reading overwriting the oldest one.

See the source list for the places to download the C code for the BMP085 and the KS0108 Graphical LCD Library. There is a PDF file (`GLCD_Documentation.PDF`) in the `DOC` subdirectory of the unzipped GLCD library. This contains a correct wiring diagram for the display; use it instead of the HTML version which contains errors.

Due to the use of the `WIRE` library for the BMP085 – which uses Arduino pins A4 and A5 – the `EN` port of the GLCD needs to be changed from A4 to something



CODE SOURCES

C code for BMP085:

<https://code.google.com/p/bmp085driver/downloads/list>

Graphical LCD Library:

<https://code.google.com/p/glcd-arduino/>

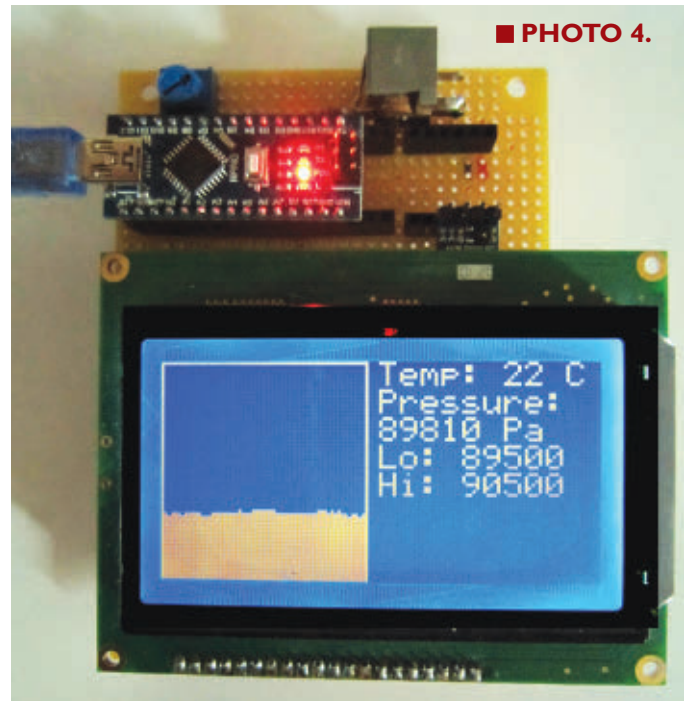
Nuts & Volts
II CD-ROMs
& Hat Special!
 That's 132 issues.
 Complete with supporting
 code and media files.

Nuts & Volts Magazine

Free Shipping!

Only \$229.95
 or \$24.95 each.

Call to order at 1-800-783-4624
or go to www.nutsvolts.com



■ PHOTO 4.

else; I picked pin 12. This change is reflected in the KS0108_Arduino.h file at the article link. Use this file to replace the one in /Arduino/libraries/glcd/config (which will be found in the Documents folder on a Windows machine). Or, you can make this edit yourself.

Be aware that not all graphical LCDs work with the GLCD library cited in the sources. Check the pictures of the pins: there should be CS1 on 15 and CS2 on 16.

The version in **Photo 3** uses an Arduino Nano. It is also using a BMP180 — an updated version of the barometric sensor. All these are functionally the same as those mentioned previously.

ITEM

Arduino Development Board:

Uno or Nano

arduino.cc

www.sparkfun.com/products/11021

www.adafruit.com/products/50

KS0108 Graphical LCD

www.sparkfun.com/products/710

www.adafruit.com/search?q=ks0108&b=1

BMP085 or BMP180 Barometric Pressure Sensor

www.sparkfun.com/products/11824

www.adafruit.com/products/1603

10K Potentiometer

www.digikey.com/product-detail/en/3362P-1-103LF/3362P-103LF-ND/1088412

220 ohm Resistor

PARTS LIST



■ PHOTO 5.

April 2015 **NUTS AND VOLTS** 27

BUILD THE SUPER KISS TIMER

By Frank Muratore

Post comments on this article and find any associated files and/or downloads at www.nutsvolts.com/index.php?magazine/article/april2015_Muratore.



■ SuperTimer (front view).

Have you ever built a project that needed a load to be switched on or off at regular intervals? Or, do you have a use for an accurate timer to facilitate switching of lights or another device with repetitious on/off cycles? Don't have the money or time to buy or build such a device? Then, the "Super Timer" is for you.

This timer not only requires a minimum parts count, but can be built in one evening. The timer is programmed via thumbwheel switches and – besides having eight modes of operation – it can run from milliseconds to 9,999 hours. (That's over a year!) The unit is built around the Omron H3CA-A timer module.

Caution: This project uses 120V AC line voltage in its operation. Only persons qualified and knowledgeable in safe handling practices should build it. If that's not you, get help from someone with proper experience before attempting to build it.

Other features are:

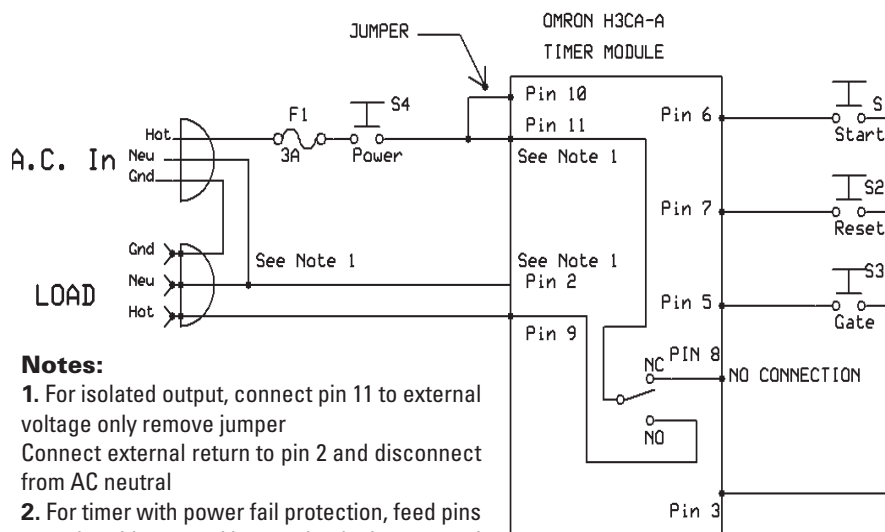
- Wide input voltage range
- 24-240 volts AC or 12-240 volts DC
- Three amp output contact rating
- Isolated supply from output
- Industrial reliability
- Compact size
- Few external parts needed

After building the Super Timer, I got the idea to simplify it even more while retaining all of the features of the original. The Super KISS (you know, for Keep It Simple Stupid) Timer was born. This newer module has a ton of versatility (see the sidebar).

Referring to the **schematic**, you can see there are only four switches required: two momentary normally open, and two alternate action switches. The momentary switches control the start and reset functions, while the alternate action switches control the power and gate functions. Also, you can eliminate the gate switch if you don't have a need to "halt" the timer operation.

Wiring is straightforward. All AC splices are soldered and protected with heat shrink. Also, strain reliefs are used (as dictated by the NEC). The code reads: "Connections must be done in a manner to not pull loose while protecting the wires from damage." This is done (very nicely) by the use of Arlington low profile strain reliefs (www.aifittings.com).

To further simplify wiring, I used a store-bought line cord and cut it in two (saving the male end for use). I took the remaining pieces and cut off the female end, then



Notes:

1. For isolated output, connect pin 11 to external voltage only remove jumper
Connect external return to pin 2 and disconnect from AC neutral
2. For timer with power fail protection, feed pins 10 and 2 with external battery-backed power and disconnect pin 10 to 11 jumper.
Feed load power to pin 11 and load neutral

■ **Timer schematic.**

added my store-bought female connector (heavy duty type).

You can buy an extension cord (of the proper size) and use both ends (if you like) to make construction even simpler. I prefer the heavy duty female for "knock-around" resistance. The other ends are wired per the schematic.

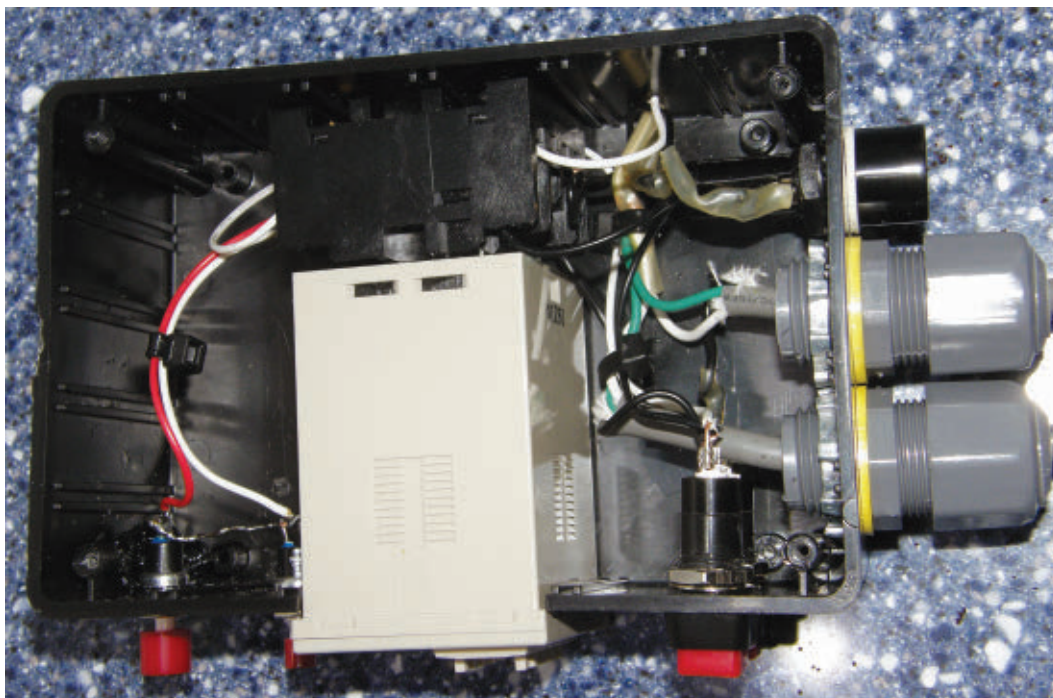
Also as shown in the **schematic**, you can wire the timer for power fail back-up. However, this requires a separate source of back-up power. For simplicity, I have opted not to do this. Also, if the power-fail option is



■ **AC cable ends.**

DESCRIPTION	PART #	SUPPLIER
Timer Module	H3CA-A	aliexpress.com
S1, S2	275-1547	RadioShack
S3, S4	275-617	RadioShack
Socket	PF113A-E	Omron
Case	270-1806	RadioShack
Fuse Holder	270-367	RadioShack
Fuse	2700150	RadioShack
Line Cord	PC-303R	CablesOnline.com
3 Prong Socket	93687	Harbor Freight
Strain Relief	LPCG503	AG Electrical
Misc. Hook-up Wire	278-1222	RadioShack

PARTS LIST



■ Inside wiring.

employed, the “start time” on the power-up function will not apply.

So, what are you waiting for? Warm up your soldering iron and get busy! By the way, another great use for the timer is to pulse a cheap soldering iron and make it temperature controlled (like the costly pro models). Just adjust the pulse rate time and you will have variable temperature. **NV**

Operating Instructions

Caution: Do not change the thumbwheel switches with power applied. Also, do not add or remove a load with power applied.

Operating Modes

MODE SELECTIONS

There are eight different operating modes from which to choose. Press the leftmost thumbwheel switch to select the desired operation mode. When making your choice, the operation mode will show in the operation mode display window. The eight operation modes are:

- A ON-delay
- B Repeat (50% fixed duty cycle)
- C Signal Interval/OFF-delay
- D Signal OFF-delay (I)
- E Interval
- F Cycle One-shot
- G Signal ON-delay/OFF-delay
- H Signal OFF-delay (II)

Mode A ON-delay (Power ON Start/Power OFF Reset): Connect start terminals 3 and

6. Upon application of power to the timer, time delay period begins. At the end of the time delay period, output contacts the switches, either connecting or disconnecting the load. Output remains switched until power is removed or a reset input is applied.

Mode A ON-delay

(Signal Start): Power is applied continuously. The time delay period begins at the leading edge of the start input. Output contact switches when the accumulated time equals the set time. Subsequent start signals during or after timing will not be accepted. The output relay will remain switched until a reset input is applied or power is interrupted.

Mode B Repeat Cycle — Signal Start (50% fixed duty cycle):

Power is continuously applied. The OFF/ON cycle is initiated at the leading edge of the start input. The output relay will be OFF for the set time and ON for the set time. The ON and OFF cycle will continue to alternate until a reset input is applied or power is disconnected.

Mode B Repeat Cycle — Power ON Start/Power OFF Reset (50% fixed duty cycle): Connect start terminals 3 and 6. Upon

application of power to the timer, the OFF delay is initiated for the set time and then ON for the set time. The ON and OFF cycle will continue to alternate until a reset input is applied or power is disconnected.

Mode C Signal Interval/OFF-delay:

Power is continuously applied. Time delay period begins on both the leading and trailing edges of the start input. Output contact switches during time delay period, either connecting or disconnecting the load. Once the timer has timed out from the trailing edge, it resets and is ready for subsequent start inputs.

Mode D Signal OFF-delay (I):

Power is continuously applied. The output relay switches at the leading edge of the start input, either connecting or disconnecting the load. Time delay period begins at the trailing edge of the start input. Output relay switches again when accumulated time equals the set time.

Mode E — Interval

Signal Start: Power is applied continuously. Timing begins at the leading edge of the start input. The output relay is switched, either connecting or disconnecting the load only during timing. The timer is



■ Module with socket.



■ Strain reliefs.



■ Timer module.

reset when power is disconnected or a reset input is applied.

Mode F Cycle One-shot Power-ON Short/Power-OFF

Reset: Connect start terminals 3 and 6. Upon application of power to the timer, timing starts. The output relay is OFF for the set time and then ON for the set time for one cycle only. The timer is reset when power is removed or a reset input is applied.

Mode F Cycle One-shot

Signal Start: Power is continuously applied. The OFF/ON cycle is initiated at the leading edge of the

start input. The output relay will be OFF for the set time and then ON for the set time for one cycle only. The timer is reset when power is removed or a reset input is applied.

Mode G Signal ON-delay/OFF-delay:

Power is continuously applied. Timing begins on both the leading and trailing edges of the start input. The output relay switches when the accumulated time from the leading edge equals the set time, either connecting or disconnecting the load. It also switches for the set amount of time from the trailing

edge of the start input.

Mode H Signal OFF-delay:

Power is continuously applied. Timing begins at the trailing edge of the start input. The output relay is switched only during timing.

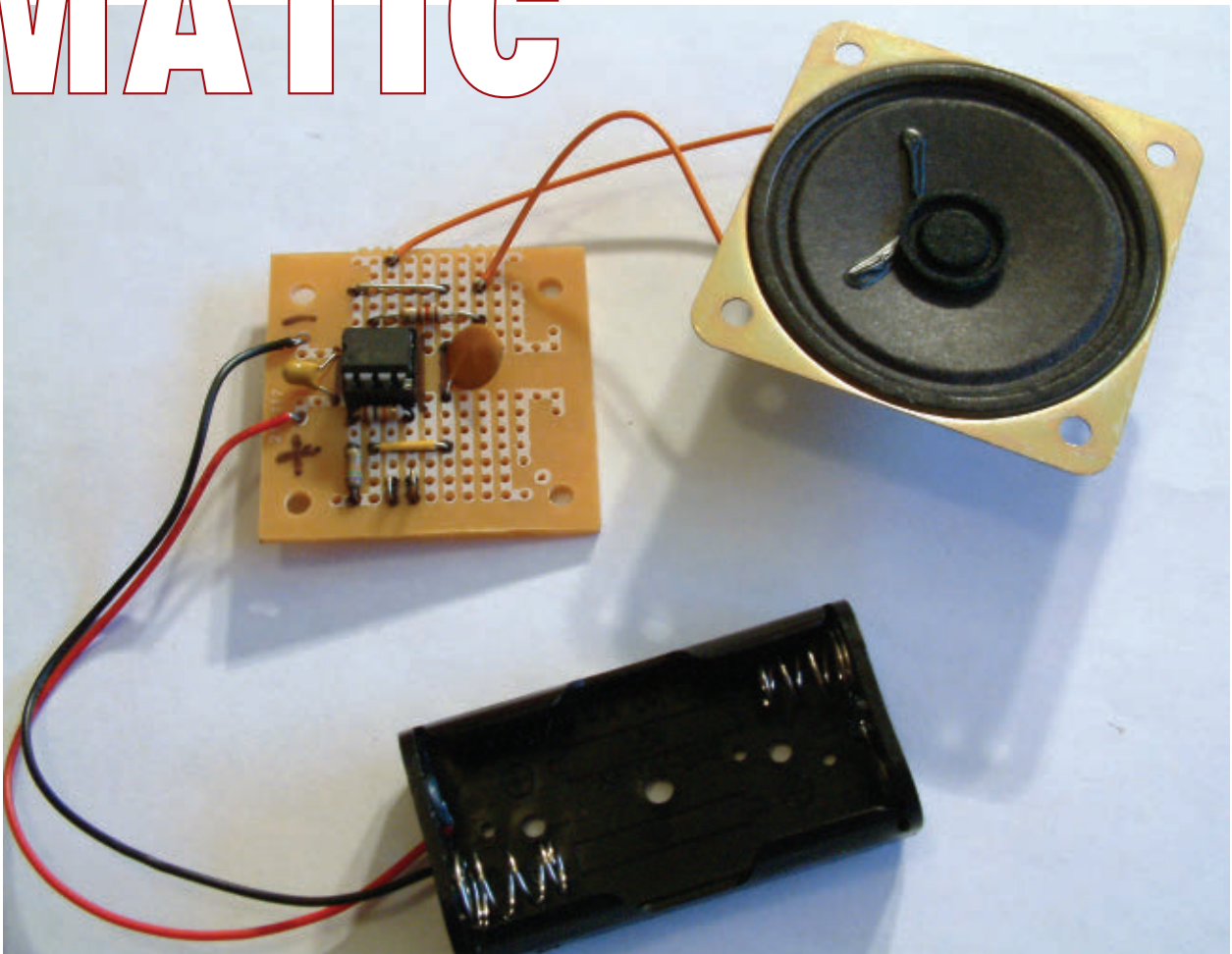
Note: All timer instructions and operating modes were taken from www.farnell.com/datasheets/1754059.pdf.

(Copyright Premier Farnell UK Limited)

By Bob Diaz

Post comments on this article and find any associated files and/or downloads at www.nutsvolts.com/index.php?magazine/article/april2015_Diaz.

BUILD THE ANNOY-O- MATIC



For years, engineers and technicians have built annoying little circuits that are easily hidden, that beep, or chirp. The AOM (Annoy-O-Matic) brings this practical joke to a whole new level. By using the PIC12F629, this project not only beeps, but the beep cycle appears to be completely random — maybe once every three to eight minutes. Despite its sophisticated operation, it's very easy for any beginning electronics student to build. Even experienced builders will find this to be a fun little project.



In designing the AOM, my key criteria was to keep it very simple. The AOM uses a minimum of parts; take a look at the **schematic** and **Parts List**.

Like many other PIC projects I've designed, the real power to the AOM is found in software. The AOM uses an external RC clock, R1 and C2, for an external clock frequency of around 32 kHz. The low frequency clock keeps current consumption to a minimum — 168 μ A to 90 μ A depending on battery voltage — and allows a fresh set of AAA alkaline batteries to last at least three months.

Microchip does not provide a lot of information as to suggested RC values for a 32 kHz external clock (8 kHz instruction clock), and after testing the AOM with different voltages, I now understand why. It seems that the external RC clock is very sensitive to changes in the supply voltage. Refer to **Table 1** for the measurements I made.

For other projects, this radical change in the clock frequency would be unacceptable. For the AOM, however, this change adds to the intensity of the annoyance.

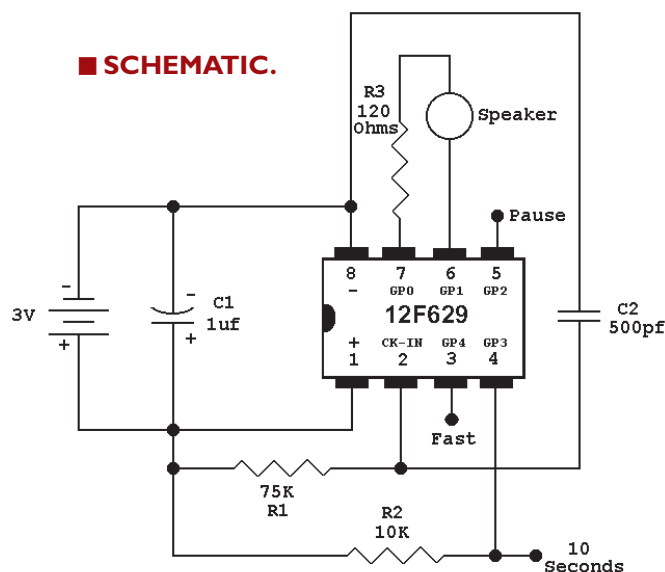
For example, when the batteries are fresh, the initial tone is around 2 kHz every three to eight minutes. When the batteries are near the end of their life, the tone is around 3 kHz every two to five minutes.

CONSTRUCTION NOTES

I found the Datak Experimenter's protoboard with standard IC and component spacing (#12-607) to be ideal for building this project. If you can't find this exact board, any small 1.8" x 1.8" standard IC protoboard will do.

Before soldering, start by drawing the parts layout on the component side of the board; see **Figure 1**. As tempting as it is to skip this step, by doing it you'll avoid the aggravation of making a mistake and having to unsolder and re-solder components back into the correct

■ SCHEMATIC.



place. Just be sure to double-check your work against the **schematic** and **Photo 1**.

C1 is a tantalum capacitor; the shorter lead is connected to the negative of the power supply. Connect this capacitor as close as possible to the IC power supply pins 1 and 8. This capacitor removes any noise spikes that might reach the PIC. Also, do NOT substitute an electrolytic capacitor. A tantalum capacitor is the best choice because it can deal with the sudden demands for current. The electrolytic is the worst choice because it has the slowest response for abrupt current demands.

For all prototypes I constructed, I used a ceramic capacitor for C2. However, I see no reason why any type of 500 pf capacitor or other values couldn't be used. (See "Operation & Design Notes.")

Supply Voltage vs. Instruction Clock

Voltage	Instruction Clock
5.5V	5,395 Hz
5.0V	5,715 Hz
4.5V	6,132 Hz
4.0V	6,680 Hz
3.5V	7,471 Hz
3.0V	8,671 Hz
2.5V	10,655 Hz
2.2V	12,543 Hz

■ TABLE 1.

C1	1 μ F Tantalum Capacitor
C2	500 pF Capacitor, any type (see "Operation & Design Notes")
R1	75K Ω , 1/4W \pm 5% recommended
R2	10K Ω , 1/4W
R3	120 Ω , 1/4W

Speaker 48 Ω or higher. I suggest using All Electronics CAT# SK-63, 2-1/4" 63 Ω speaker (\$1.25 each)

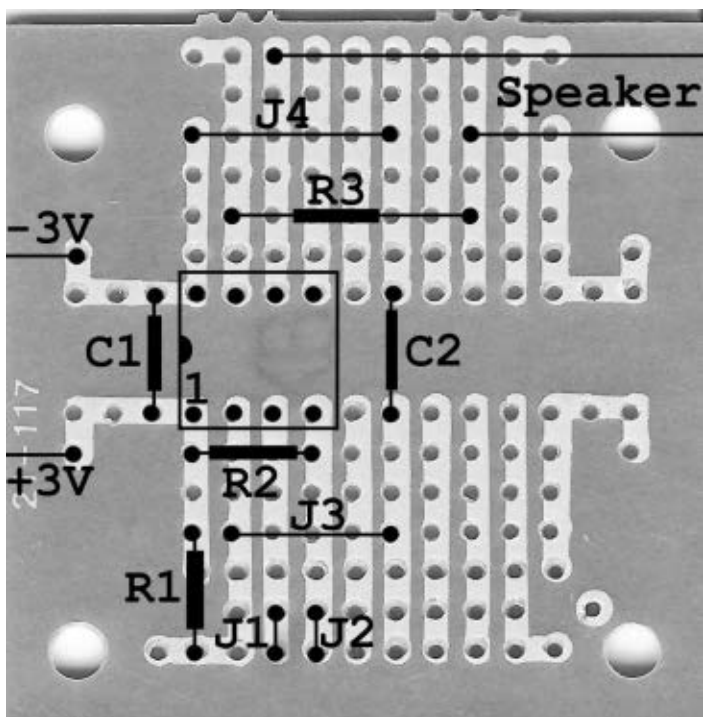
PIC12F629 DIP, plastic (PIC12F675 may be used as a replacement). You will need to program the PIC.

Eight-pin DIP IC Socket

IC Protoboard, Datak #12-607 recommended

AAA, Two-cell Battery Holder or AA Two-cell Battery Holder

PARTS LIST



■ **FIGURE 1.**

R1, R2, and R3 are 1/4W resistors; higher wattage resistors can be used if needed. R1 sets the timing for the RC clock. While not super critical, I suggest $\pm 5\%$ tolerance. R2 and R3 values are not that critical, so even if they were off by $\pm 20\%$, the circuit will still work.

The best speaker I found was a small 63Ω unit from All Electronics. The small size and low cost offered the

loudest sound for the money. Larger 48Ω speakers also work, but were overkill for my wishes. Using a lower impedance than 48Ω is possible, but the volume is reduced.

In earlier versions of the AOM, I did use 48Ω mini transducers (All Electronics CAT# PE-52), but even with two in series, the sound was not loud enough for my taste.

Standard AAA alkaline batteries should last for at least three months. If you wish to use standard AA alkalines, the minimum running time would be increased to at least six months. I chose two AAA batteries in a holder because they were small and easy to conceal. Still, even AA batteries aren't so large that the unit couldn't be hidden somewhere.

The IC is the PIC12F629, but the PIC12F675 will also work in its place. I strongly recommend using an IC socket. Should you wish to reprogram the chip, it's easy to remove it.

Pins 3, 4, and 5 of the IC allow for special test features and an additional function for the AOM. Pins 3 and 5 make use of the internal weak pull-up resistors inside the PIC. Pin 4 uses the external pull-up resistor, R2. While you don't have to wire any connecting wire or post to these pins, I recommend wiring at least one wire to pin 4. Once built, this pin allows for quick and easy testing.

TESTING

The test/feature pins are defined as follows:

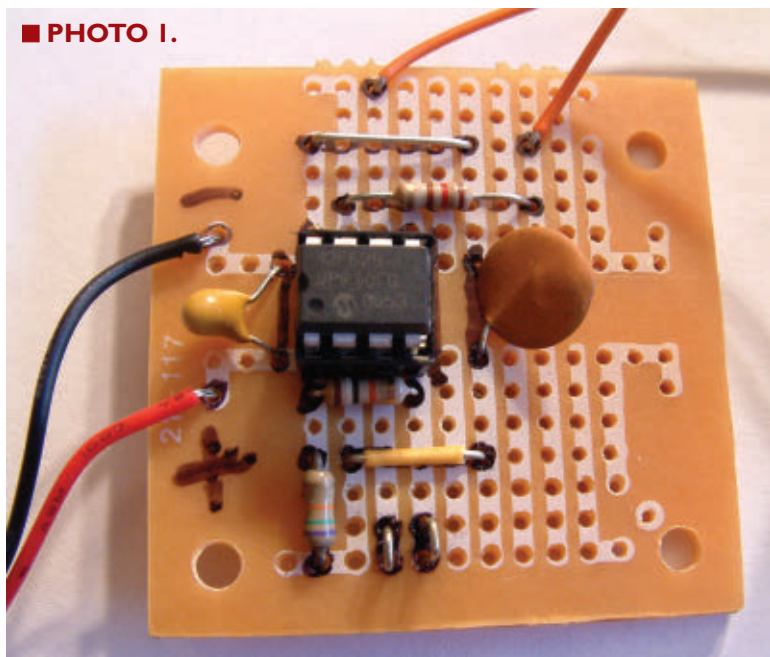
Pin 3 GP3 Fast: When connected to ground, it shortens the timing cycle by around 1/120 of the time. This causes the unit to beep every few seconds (random time period); (J1).

Pin 4 GP3 10 Sec: When connected to ground before power is applied, the AOM beeps around every 10 seconds; (J2).

Pin 5 GP2 Pause: When connected to ground, this pauses the time count; (NC).

Once built, you could install the batteries and wait roughly three to eight minutes before you might hear a beep. Both pins 3 and 4 allow you to shorten the time you need to wait.

There are two choices for speeding up the beep cycle: pin 3 and pin 4. Pin 4 (J2) — the 10 second cycle — provides the most information about the circuit. Remember, this pin must be grounded before the power is applied. Once the power is applied, if you measure the time between each beep, you'll have a reasonable estimate for the instruction clock frequency, the initial beep frequency, and the RC clock frequency.



■ **PHOTO 1.**

$83,000 / \text{Seconds Between Each Beep} = \text{Instruction Clock}$

This should give you a reasonable estimate of the instruction clock's frequency. With fresh batteries, it should be close to around 8 kHz. As the batteries age, the frequency increases.

It takes four instruction cycles to produce a single cycle of beeps:

$\text{Instruction Clock} / 4 = \text{Beep Frequency}$

The RC clock is always four times greater than the instruction clock. So, multiply the instruction clock frequency by four to determine the RC clock frequency. With fresh batteries, it should be close to 32 kHz.

Grounding pin 3 (J1) shortens the time delay by roughly 1/120 of the time. This provides a quick way to hear the random nature of the beeping without waiting too long.

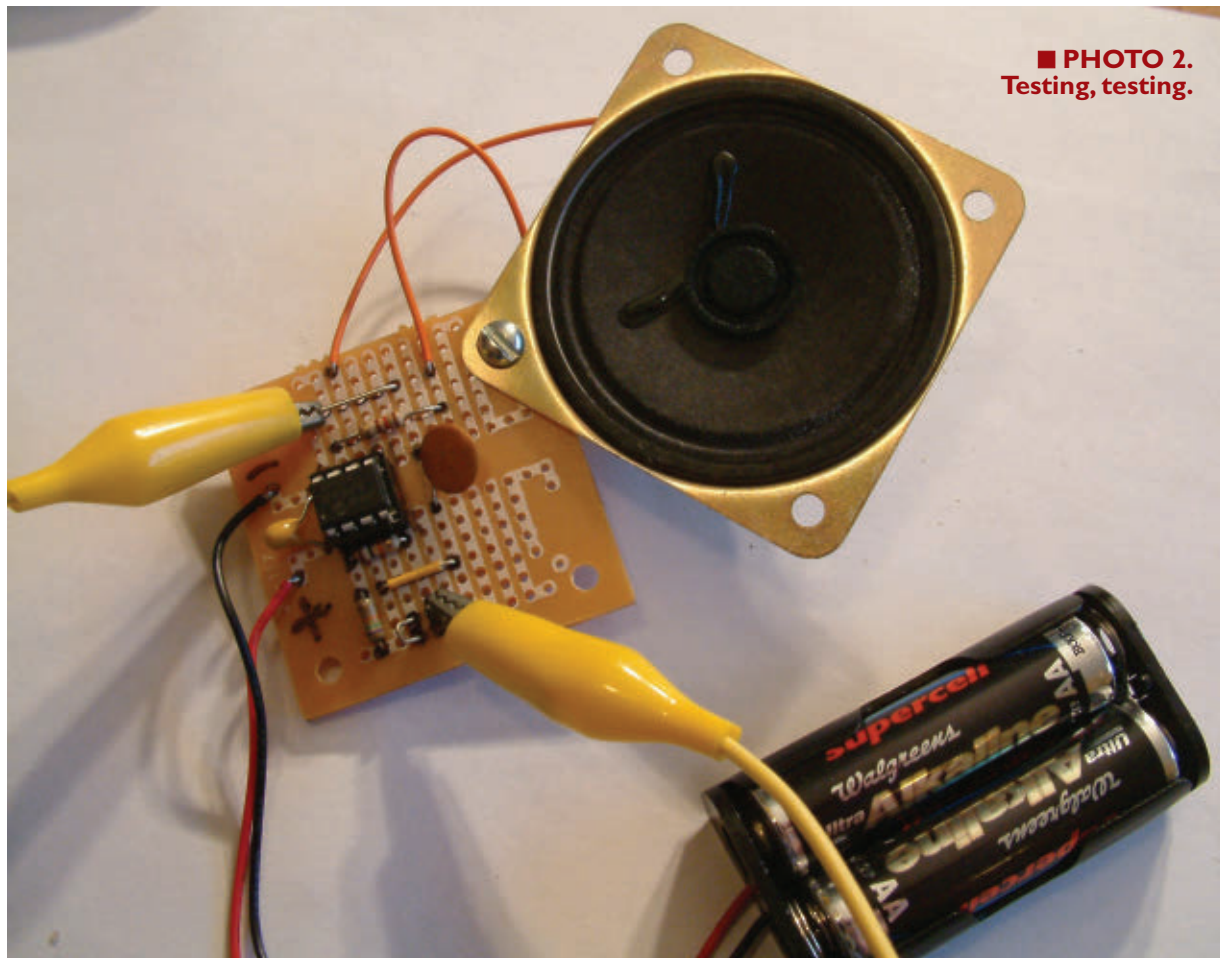
Grounding both pin 3 (J1) and pin 4 (J2) generates a rapid string of beeps. This proved to be ideal for measuring and testing the inductive kick in the speaker. More on that in "Operation & Design Notes."

Pin 5 (Pause) is for possible expansion. While I have not designed any circuit or tried anything yet, one could attach a phototransistor or some sort of photodetector circuit to this pin. Thus, when the lights are on, no beeps occur; when it's dark, the beeping resumes. This might be perfect in a bedroom where the victim only hears the beeping when it's dark. If you chose to explore this type of operation, you may want to shorten the time between the random beeps.

OPERATION & DESIGN NOTES

The one area I spent the most time testing and exploring was the speaker. Using a 120Ω resistor for R3 is a very conservative approach to the design. This allows you to use any impedance speaker without drawing too much current from pins 6 and 7.

The All Electronics (CAT# SK-63) 2-1/4" speaker is roughly 63Ω resistance and around 1.5 mH of inductance. Even under worst case conditions — an output voltage of 3.6V and the frequency of the tone is 1.5 kHz



■ PHOTO 2.
Testing, testing.

Pseudo Random Number Generation

Most software pseudo random number generators simulate a shift register with taps at different points. The taps are XORed and form the input to the shift register.

The other approach is to use a look-up table to read random numbers. Due to the limits inside the PIC's memory, a 256 byte random number look-up table was about the biggest I could create. This sequence would repeat roughly every day.

In order to make the table look bigger, I have two pointers moving through the table. The first pointer moves forward through the table reading each number, and the second pointer moves backwards through the table reading each number. To generate a difference after 256 bytes, the second pointer does not move when the first pointer is at position zero. In addition, in order to generate a different sequence of numbers from the second pointer, the upper and lower nibbles are swapped and some bits are inverted on the second pointer's output.

Both pointers return two different random numbers, where one random number is subtracted from the other. The end result is a string of eight-bit numbers that does not repeat until 65,536 numbers later. I'm sure a cryptographer could find a pattern to this sequence, but the average person will have no idea what the next number will be.

— R3 would still limit the current to under 25 mA.

Another area of concern was the inductive kick from the speaker. Even with a three volt supply, I measured

around five volts peak from some speakers. This high a voltage above VDD might damage the chip.

The normal approach to this would be to place a diode (like 1N914) between the positive power supply (pin 1, VDD) and pin 6, as well as another diode between pin 7 and pin 1. The diodes would be oriented to conduct whenever pin 6 or 7 is a higher voltage than the positive power supply; thus removing any spikes.

The only problem with this solution was that many of my beginning electronics students frequently solder diodes into circuits backwards because they don't fully understand what they are doing. This raised the question, "Are the protection diodes really necessary?"

In order to answer that question, both pins 3 and 4 were grounded to make the AOM output a series of rapid beeps. The unit was tested for three weeks with supply voltages ranging from 2.2V to 5.5V. After millions of beeps — roughly 1,000 times more beeps than the unit will produce in a month of normal operation — my AOM continues to work without problems.

For those who would like to modify the design to shorten the delay times and increase the pitch, change C2 from 500 pF to 330 pF. This would cause the pitch of the tone to start at 3 kHz and be 5 kHz by the end of the battery's life. The delay times would range from two minutes to five minutes with fresh batteries, and two to three minutes by the end of the battery's life.

Values larger than 500 pF result in a slower instruction clock with a lower pitched tone and longer delays. Values less than 500 pF result in a faster instruction clock with a higher pitched tone and shorter delays. For those who wish to try things out, try values for C2 from 1,000 pF (0.001μF) to 200 pF.

Another area for modification is changing R2 to be a 5KΩ resistor in series with a 100KΩ potentiometer. An increased resistance results in a slower instruction clock, and lower resistance results in a faster instruction clock. For the PIC's RC clock, R2 must be a minimum of 5KΩ, to a maximum of 100KΩ.

Should you settle on a faster instruction clock, remember that a faster clock results in increased current consumption. If you chose to use a 330 pF capacitor for C2 or reduced resistance for R2, you may wish to use the larger capacity AA batteries rather than the lower capacity AAA batteries.

Part Sources

All Electronics 14928 Oxnard Street Van Nuys, CA 91411	818-997-1806 www.allelectronics.com
--	--

Digi-Key 701 Brooks Avenue South Thief River Falls, MN 56701	218-681-6674 www.digikey.com
--	--

Jameco Electronics 1355 Shoreway Road Belmont, CA 94002	650-592-8097 www.jameco.com
---	--

Mouser Electronics, Inc. 1000 North Main Street Mansfield, TX 76063	817-804-3888 www.mouser.com
---	--

Datakit Protoboard Information:
www.philmore-datakit.com/protoboards.htm
www.philmore-datakit.com/DatakitDist.htm

Closing Comments

Over the last 1-1/2 years, I've had roughly 100 students build this project. Based on their suggestions, several variations were tried and minor improvements to the design were added.

One of the best suggestions was to generate a multi-tone chirp. If you look at the source code available at the article link, you'll see that a chirp starts at 2 kHz and then

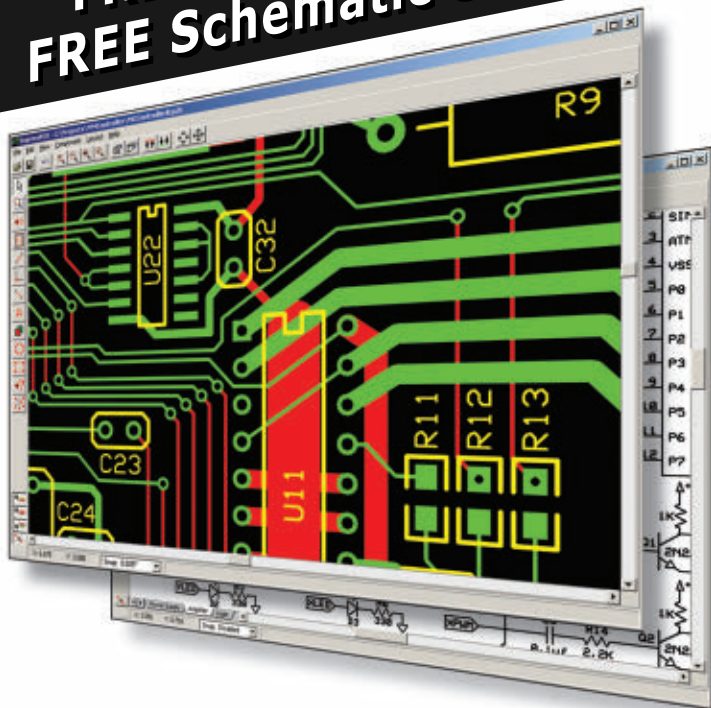
shifts to 1.5 kHz. This shift in the frequency makes it sound almost like a dripping water facet.

For those who want to experiment, not only are there several changes to the hardware you can make, the source code can be adjusted as well to change the timing or the sound. All the files, including

the .HEX file to program the PIC, can be downloaded from the article link.

Last of all, putting the AOM inside a box would help protect the components and provide a better sound from the speaker. Just make sure there's a large enough hole for the sound of the speaker to be heard clearly. **NV**

\$51^{For 3} PCBs
FREE Layout Software!
FREE Schematic Software!



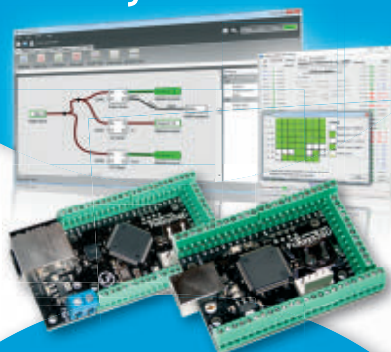
- 01 DOWNLOAD our free CAD software
- 02 DESIGN your two or four layer PC board
- 03 SEND us your design with just a click
- 04 RECEIVE top quality boards in just days

expresspcb.com

Imagine this...

- a small and inexpensive
- USB or ethernet device
- with a rich set of features
- including PLC functionality and 3-axis stepper motor controller
- all accessible via free .NET, ActiveX or C++ library
- cross-platform
- configurable with free software

PoKeys



Or this...

- all in one: Oscilloscope, Data Recorder, Logic analyzer, Analog and digital signal generator
- smallest USB 2.0 portable 1MS/s oscilloscope
- data acquisition of analog and digital signals
- data recording
- export to CSV, XLS, PDF and HTML
- simple usage of advanced features
- examples for C++, VB, Delphi and LabView
- free software and updates

PoScope Mega 1+



PoLabs



Visit us at
www.poscope.com

Beyond the Arduino 2

Beyond the Arduino IDE

We're exploring the world of working directly with AVR microcontrollers. In the first installment in this series, we built our own simplified Arduino Uno on a breadboard. Now that we've established a hardware platform to work on, we're going to dive into working in a new environment. Mask and snorkel on!

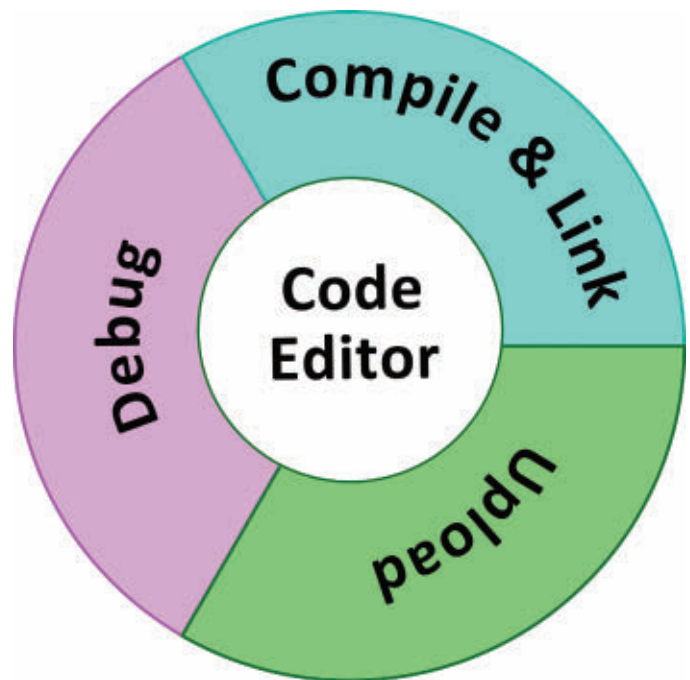


FIGURE 1: Components of a typical IDE.

All Aboard!

The journey to working efficiently with the Atmel AVR range of microcontrollers is an exciting one. That is, if you enjoy learning new ways of doing things, gaining greater control over the microcontroller, working smarter and more efficiently, and doing things you weren't able to do before! Convinced?

Last month, we looked at a few good reasons to start working with the raw AVR microcontroller, and then built our own breadboard-based Arduino Uno. We'll be using that breadboard project this month as we get stuck into working in a new IDE.

What's in an IDE?

An IDE — or Integrated Development Environment — is where you spend most of your time working on embedded projects. An IDE would normally include the code editor, a compiler, a linker, and an uploader to Flash the code onto the microcontroller (**Figure 1**). More advanced IDEs also contain debuggers. The compiler,

assembler, and linker are called a toolchain — the most popular for AVR microcontrollers is GCC (see [1] in **Resources**).

In my software developer past, the IDE was a critical tool in getting me working efficiently — I'd make sure I configured the IDE so that it suited the way I liked to work. If you've worked in other environments, you'll know exactly what I mean.

The Arduino IDE is a great tool to get you started with embedded systems, but it is very basic as IDEs go. The strength of the Arduino environment lies in the way in which some fairly complex functionality is abstracted away from the designer in a number of libraries — you'll see as we work through this series how functionality like reading an analog value is made much simpler through the Arduino IDE.

I can sense that some readers are questioning why we're trying to make things less simple. The answer is that we're trading simplicity for increased functionality and flexibility. Stick with me and you'll see the benefits. I had a number of false starts before I put my head down and tackled it head on. I haven't looked back!

Good Bye Arduino IDE, Hello ... ?

Do a quick search and you'll see that there are a number of AVR IDEs to choose from. The more popular ones are Atmel Studio, Eclipse with the AVR plug-in, and IAR's Embedded Workbench. In fact, you could even stick with the Arduino IDE and just ignore the Arduino libraries. The *Popular IDEs* sidebar gives an overview of some of the features of these more popular IDEs.

Personally, I prefer to use Atmel Studio. It is developed by Atmel (the manufacturer of the ATmega328P on our breadboard Arduino), so I know it'll support all my AVR microcontrollers without any problem. Unfortunately, it is only available for Windows, but it is dead easy to install and use, has a good set of functionality, and is free.

Along with the new IDE comes a host of new ways to interact with the microcontroller — you'll be working at a lower level without the abstraction that the Arduino IDE provided. While this is a little more complex, it gives you loads more control and flexibility, and is one of the key reasons to make the move.

What? No More *digitalWrite*?

That's right! We've left our *digitalWrite()*, *pinMode()*, *digitalRead()*, and a whole bunch of other functions behind. These functions concealed some of the low-level instructions needed to access the microcontroller's functionality — so we now need to get down and dirty to do this ourselves. Before we roll up our sleeves, let's get a couple of principles out the way.

Pin Numbering

In our Uno world, we had 14 digital pins (D0–D13) and six analog pins (A0–A5). This numbering system was developed by the team at Arduino to make it easier to work with the inputs and outputs. In reality, the microcontroller I/O pins are divided up into “ports” and “pins.”

Pins are the physical I/O pins on the microcontroller, excluding the various power and ground pins. To make it simpler to access the pins, they are grouped into ports; usually eight pins per port. For Atmel controllers, pins are numeric (0–7) and ports are alpha (starting at A). Each pin is

Popular IDEs

ARDUINO IDE

Many hobbyists stick with the Arduino IDE when they move onto using stand-alone microcontrollers — just as we did in the first article. The Arduino IDE offers a familiar environment and runs on multiple operating systems easily, but has an extremely limited set of features. In addition it does not support a wide range of AVR microcontrollers “out the box.” I wouldn't suggest it as an option.

ATMEL STUDIO

Atmel Studio is — as its name suggests — Atmel's own in-house IDE. It leverages off Microsoft's Visual Studio platform, and therefore only runs on Windows operating systems. It does, however, offer a number of benefits to the hobbyist and enthusiast, which I believe outweigh the platform limitations. For a start, it's free, without any restrictions on the size of code you can write. It also uses the GCC toolchain, which is the de facto open source standard for AVR microcontrollers. A big attraction for me was the ease of installation and built-in support for all Atmel's microcontrollers, not just the eight-bit AVR. So, if you move onto using Atmel's other series such as the ARM, you will work in the same IDE. Finally, it is pretty full-featured: it uses Microsoft Intellisense to make coding faster; has a built-in simulator to test code before it even sees a microcontroller; and has a debugger to allow you to debug code as it's running on your microcontroller (see [2] in **Resources**).

IAR EMBEDDED WORKBENCH

IAR Embedded Workbench for AVR is a professional-level development tool — and comes at a professional price! You do have the option to use a code-size limited version, but as soon as your code exceeds 4 Kb in size you need to pay up or move on. It also only runs on Windows. It is a fully-featured IDE that allows simulation and debugging. A key advantage is that IAR has products for many other manufacturers, so if you can manage the price it will allow you to work in one environment for most of the microcontrollers you're likely to use (see [3] in **Resources**).

ECLIPSE WITH AVR PLUG-IN

If you are a supporter of multi-platform open source software, then the Eclipse IDE with the AVR plug-in is perfect for you. If you don't like performing multiple downloads and complex installation processes, then this is NOT perfect for you. Eclipse is a really great IDE and has a strong community behind it; the AVR plug-in, however (and I'm sure I'll get shot down by someone), seems to have stagnated since the end of 2011. If you're feeling brave, there are a number of tutorials out on the web that should get you up and running without too much pain (see [4] and [5] in **Resources**).

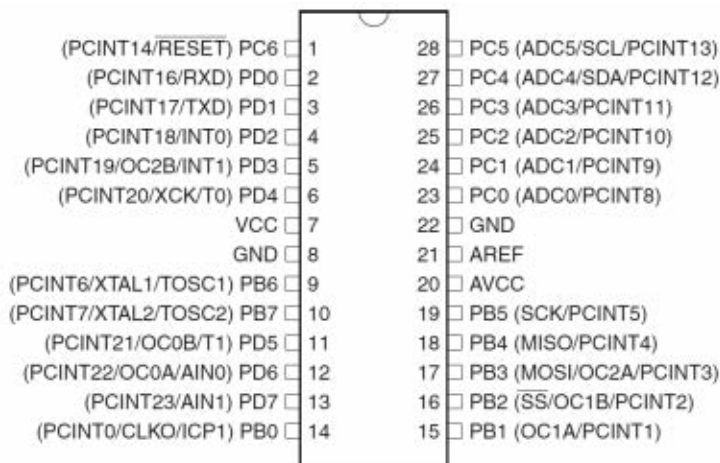


FIGURE 2:
ATmega328P pinout diagram from the datasheet.

therefore referenced by its port and pin number, prefixed with a “P” (e.g., PB1, PB0, PD7). **Figure 2** is from the ATmega328P datasheet, and shows how the individual pins are named — you’ll notice that they don’t flow logically from the first to the 28th pin, not all ports have eight pins, and there’s no PORTA.

When I first looked at the pinout diagram in the datasheet, I nearly flipped! There was so much complexity and very little to explain what all the acronyms in parentheses were. Don’t panic! We’ll tackle these when we need them over the course of this series.

Registers

Okay, so we know that each of the I/O pins are numbered differently to the Arduino IDE and how they’re numbered. How do we read from or write to them? Most microcontrollers use registers to do this. A register (or more correctly, a hardware register) is a bit like a pre-defined variable — you can read the value stored in the register and change it (if it’s not a read-only register).

In the background, each hardware register is linked to specific hardware-related functionality, and writing to them causes the hardware to behave in certain ways. Registers have specific memory addresses, but we usually refer to them using the “friendly” names that are defined in the datasheets and header files.

The simplest hardware registers are simply called “PORT” registers. There is one of these registers for each of the ports on the MCU. For the ATmega328P, there are PORTB, PORTC, and PORTD registers. By writing to these, you cause the individual pins to go either high or low in the same way as *digitalWrite()* did in the Arduino IDE.

But which pin? A port contains up to eight pins, so how does the PORT register allow us to access a specific pin? Simple! Or, so I was told when I first tackled this. However, it took me a while.

As mentioned, each port has up to eight pins. A byte has eight bits. A register is usually one byte in size. If you make the connection, you’ll realize that each bit in the eight-bit register is linked to a pin on that port. **Figure 3** shows an example of how this would look for PORTB, with PB0 and PB4 set high.

In order to make pin PB2 go high, you need to set bit 2 of the PORTB register to a 1. To make PB4 go low, you

Pin Number	PB7	PB6	PB5	PB4	PB3	PB2	PB1	PB0
Bit Number	7	6	5	4	3	2	1	0
Pin High/Low	Low	Low	Low	High	Low	Low	Low	High
PORTB Register	0	0	0	1	0	0	0	1

FIGURE 3: Register for PORTB with PB0 and PB4 set high.

need to set bit 4 of the PORTB register to 0. Behind the scenes, this is exactly what *digitalWrite()* was doing for you.

One final piece of theory before we get stuck into a first project: bitwise operations. We need to use bitwise operations in order to set the bits in the PORT to 1 or 0.

Bitwise Operations

Bitwise operations are used to manipulate individual bits, or groups of bits. We won’t go into all the uses for these operations and operators here, but will focus on what we need them for: to set specific bits in registers.

Let’s assume you have PORTB configured as in **Figure 3**; bits 0 and 4 are set to 1 and the rest to 0. In binary, this would be represented as 0b00010001 (where the prefix 0b indicates a binary number follows). Now, let’s set PB2 high (i.e., bit 2 will be set to 1).

One way to do this is by spelling it out and setting PORTB to 0b00010101. However, you can only do this if you keep track of the values of all the other bits in the register — i.e., you also need to know the values of the bits that you aren’t setting. This is not very practical in a dynamic embedded system. Bitwise operators allow us to change the value of just PB2, without affecting (or needing to know) the values of the other bits in the register. Bitwise operation is critical in working with microcontrollers.

You may have come across the bitwise operators (AND, NOT, OR, XOR), as well as the shift operators (Left-Shift and Right-Shift). I want to get us to a working project as soon as possible, so I won’t delve into the detailed theory behind these operators. What I will do is quickly summarize what you need to know to get working.

Bitwise OR

The bitwise OR operator compares two binary values and returns a value that includes the 1s from both of them. In the example in **Figure 4**, bit 2 contains a 0 in the original register, and a 1 in the value it is being OR’ed with; the result is therefore a 1. Practically, if we want to set PB2 to a 1, we simply need to perform the OR operation: `PORTB = PORTB | 0b00000100`.

Bitwise AND

The bitwise AND operator compares two binary values, and returns a value that only includes the 1s where both of the values contain a 1. In the example in **Figure 4**, we want to check whether bits 2 and 0 are set (i.e., are 1).

By ANDing a binary value of 0b00000101 (where bits 2 and 0 are 1), we can see that only bit 0 was set in the original register. In practice, this would look like `PORTB & 0b00000101`

Operator	Symbol	Used For	Example
OR	(pipe)	Setting specific bits to a "1"	<pre> 0b 0 0 0 1 0 0 0 1 OR 0b 0 0 0 0 0 1 0 1 ----- = 0b 0 0 0 1 0 1 0 1 </pre>
AND	& (ampersand)	Checking whether specific bits are set (called "masking")	<pre> 0b 0 0 0 1 0 0 0 1 AND 0b 0 0 0 0 0 1 0 1 ----- = 0b 0 0 0 0 0 0 0 1 </pre>
NOT	~ (tilde)	Combine with AND to clear a specific bit	<pre> NOT 0b 0 0 0 0 0 1 0 1 ----- = 0b 1 1 1 1 1 0 1 0 </pre>
XOR	^ (caret)	Toggling bits	<pre> 0b 0 0 0 1 0 0 0 1 XOR 0b 0 0 0 0 0 1 0 1 ----- = 0b 0 0 0 1 0 1 0 0 </pre>

FIGURE 4: Summary of the commonly used bitwise operators.

above bitwise operation examples, you're probably thinking you'll be typing 1s and 0s for the rest of your life. Thankfully, bit-shifting is here to rescue you. When I first

Bitwise NOT

The bitwise NOT operator simply switches 1s to 0s, and 0s to 1s in a binary value. It operates on a single value, and does not do a comparison between two values as the AND and OR operators do. The NOT operator is useful for setting bits to a 0 when combined with a bitwise AND.

Let's say we want to unset PB4. First, we apply the NOT to the bits we want to clear, and then AND the result to PORTB:

```

NOT 0b 0 0 0 1 0 0 0 0
-----
= 0b 1 1 1 0 1 1 1 1
AND 0b 0 0 0 1 0 0 0 1
-----
= 0b 0 0 0 0 0 0 0 1

```

(Bit 4 is a 1 as this is the bit we want to unset.)

(Result of the NOT operation. Now, we AND it with the value of PORTB.)

(The result is that only bit 4 has been unset.)

encountered bit-shifting, my eyes glazed over and I moved on to something that seemed less complicated. The reality is that it's very simple.

Let's say you want to refer to PB4 on PORTB (PB4 is bit 4). You can either use the same format we used in our example (0b00010000) or you can "left-shift" a bit into position 5. The Left-Shift operator is a double angle-bracket pointing to the left (<<).

A few examples explain it best:

```

0b00000001 << 1 = 0b00000010
0b00000001 << 4 = 0b00010000 (This is the position of PB4)

```

As you know, the decimal value of a binary 0b00000001 is simply 1. So, to make your life easier, you can left-shift the decimal value 1 to the position of PB4 which is `1 << 4`.

Summarizing All the Theory

Let's combine all the above with a few examples.

Figure 5 shows the initial values of the PORTC register, pins 0-5, and then the changes to the pins as we perform

Bitwise XOR

Finally, we'll look at the bitwise XOR operator. This operator compares two binary values and returns 1 where the corresponding bits differ, or a 0 where they are the same. This is a useful way to toggle a bit; for example, if you want to flash an LED on and off. If you wanted to toggle PB4, you would use it like this: `PORTB = PORTB ^ 0b00010000`.

Shifting Bits

After you've looked at the

Description	\In Code	PC5	PC4	PC3	PC2	PC1	PC0
Initial Values		0	0	0	0	0	0
Set PC3 to high	<code>PORTC = PORTC (1<<3)</code>	0	0	1	0	0	0
Set PC4 to high	<code>PORTC = PORTC (1<<4)</code>	0	1	1	0	0	0
Clear PC3	<code>PORTC = PORTC & (~(1<<3))</code>	0	1	0	0	0	0
Toggle PC3	<code>PORTC = PORTC ^ (1<<3)</code>	0	1	1	0	0	0
Toggle PC3	<code>PORTC = PORTC ^ (1<<3)</code>	0	1	0	0	0	0

FIGURE 5: Using the bitwise operators to set register pins.

Choosing a Programmer

There are a large number of programmers available for AVR devices — a reflection of the popularity of AVR microcontrollers amongst hobbyists and enthusiasts. It would be impossible to highlight them all here, so I've gone for four of the more cost-effective ones.

Atmel AVRISP mkII In-System Programmer

This is Atmel's own programmer, so is natively supported by Atmel Studio. It, of course, supports all the AVR ATtiny, ATmega, and ATXmega microcontrollers, so it's a good choice if you want something that's simple to use and does what it says on the box. It's pretty cost-effective (\$34 at time of print), and is the de facto standard for programming AVRs (see [7] in **Resources**).

USBTinyISP

The USBTinyISP is the programmer that I use. The attraction for me was that it was open source (so you could even build one yourself); it's available from a number of manufacturers in different guises (Adafruit sells them as a USBTinyISP kit, and SparkFun as a complete board called the PocketAVR Programmer); and it comes in at around \$16. The disadvantage is that they aren't supported natively by Atmel Studio, but we work around that fairly easily in this article (see [9] in **Resources**).

Atmel-ICE

This is a fairly new product from Atmel, and is, in fact, more than a programmer — it's a debugger too. This means that you can debug code while it's running on your AVR microcontroller — a very useful thing to be able to do as your projects increase in complexity. It can program and debug AVR and ARM microcontrollers, so if you're sticking with Atmel and start working on larger more powerful ARM processors, it's a good choice. The Basic version comes in at \$49, so it's not out of reach. As soon as my local supplier gets these in stock, it's on my shopping list (see [8] in **Resources**).

Arduino as an ISP

You may have come across tutorials explaining how you can use your Arduino Uno as a programmer for raw microcontrollers. The Arduino team have written a sketch for upload onto your Arduino Uno, that enables you to program AVR microcontrollers. This is a great way to start, but does take your Arduino out of circulation. Additionally, it takes a little time to connect all the correct pins. Halfway through my first project, I ditched this and spent the \$16 to buy the USBTinyISP.

the various operations shown.

Enough Theory, Please!

We've buried ourselves in a whole bunch of theory, but theory alone isn't going to get that LED blinking. I've always been the sort of person who refers to the manual only when absolutely necessary, but when I first looked at the cryptic code in Atmel Studio I went straight for the theory. So, thanks for sticking out the fundamentals. I hope it has set the scene to get that LED blinking!

Step 1: Get Prepared

The first step to get going is to download and install your IDE. If you would prefer not to use Atmel Studio, then you can still go along with these articles; you'll just

Programmer	ATmega328
MISO	18 (MISO)
5V	Breadboard positive power rail
SCK	19 (SCK)
MOSI	17 (MOSI)
RST	1 (RESET)
GND	Breadboard negative power rail

FIGURE 6: Hooking up a programmer to an ATmega328P microcontroller.

need to find the corresponding menu commands for compiling and Flashing the microcontroller. At the level we're working at, you should be fine using this code in other IDEs, preferably using the GCC toolchain.

Atmel Studio (see [2] in **Resources**) is easy to download and install — just follow the setup wizard.

Step 2: Connect the LED

We're going to be doing the same thing as we did in the first article by connecting our LED. Connect the LED and resistor in series to PB0 (that's pin 14) on the microcontroller.

Step 3: Connect the Programmer

In the previous article, we used an FTDI breakout board to program the microcontroller with our Arduino sketch. This month, we're going to step it up a notch and use a "real" programmer.

There are two advantages to this. Firstly, you don't need a bootloader on your

microcontroller which means you can use microcontrollers that don't have bootloaders written for them (the previous article discussed how bootloaders work).

Secondly, we can program any AVR microcontroller with a programmer; we can only use an FTDI breakout board for those with dedicated serial pins (more on serial later in the series).

I'm sure you've seen those six pins labelled ICSP on your Arduino Uno. They're there to connect a programmer. Another choice needs to be made here: What programmer should I use? I've highlighted a few of the more popular ones in the sidebar, *Choose a Programmer*. For now, if you're looking for a cost-effective one, I recommend one based on the USBTinyISP or USBASP. If you want one later that will allow you to interactively debug your programs, then it's probably a

FIGURE 7: The USBTiny programmer hooked up to the ATmega328P.

good idea to buy one that supports that up-front.

Connecting the programmer is straightforward, with reference to the pinout diagram in the documentation (see [6] in **Resources**). Simply hook up the pins as per **Figure 6**.

From looking at the above, you'll start to see that those cryptic labels on the ATmega328P pins actually have a meaning and a use. **Figure 7** shows what this looks like on our breadboard.

Step 4: Write the Code

Fire up your IDE and create a new "C" project. In Atmel Studio, click on "New Project" and then select a "GCC C Executable Project" (**Figure 8**). Choose where to save your project and give it a name; you'll then be asked to select which microcontroller you're working with (**Figure 9**). Based on your choice, Atmel Studio will include the relevant header file definitions for the pins, ports, etc.

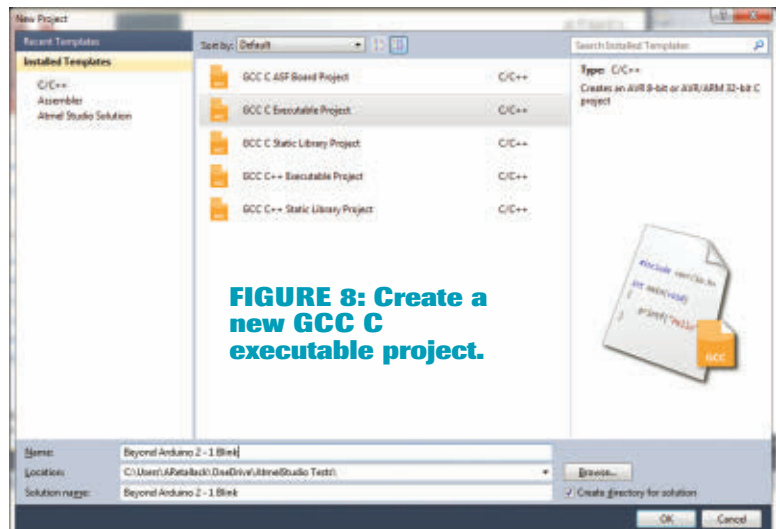
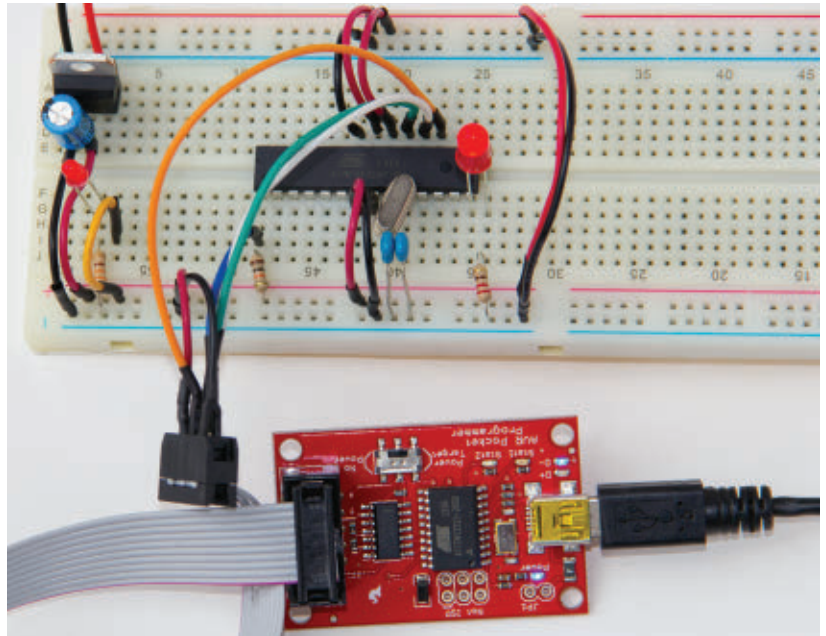


FIGURE 8: Create a new GCC C executable project.

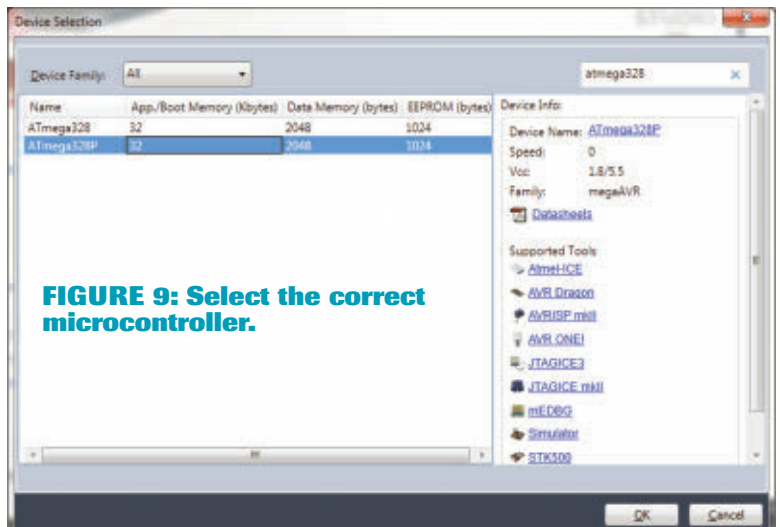


FIGURE 9: Select the correct microcontroller.

Resources

- [1] Getting started with the GCC Toolchain: www.nongnu.org/avr-libc
- [2] Atmel Studio: www.atmel.com/atmelstudio
- [3] IAR Embedded Workbench: www.iar.com/Products/IAR-Embedded-Workbench/AVR
- [4] Eclipse: www.eclipse.org
- [5] AVR Plug-in for Eclipse: http://avr-eclipse.sourceforge.net/wiki/index.php/Plugin_Download
- [6] SparkFun USBTiny Hookup Guide: <https://learn.sparkfun.com/tutorials/pocket-avr-programmer-hookup-guide>
- [7] Atmel Studio Supported Programmers: <http://store.atmel.com/CBC.aspx?q=c:100115>
- [8] Atmel Studio Supported Debuggers: <http://store.atmel.com/CBC.aspx?q=c:100112>
- [9] SparkFun's USBTiny-based Pocket AVR Programmer: www.sparkfun.com/products/9825
- [10] AVRdude Download: <http://download.savannah.gnu.org/releases/avrdude>
- [11] Author's website: www.crash-bang.com

```

/*
 * Beyond Arduino 2 - 1 Blink.c
 *
 * Nuts & Volts - Beyond Arduino #2
 * -----
 * Blinks an LED on PB0, illustrating register
 * usage and bitwise operations
 *
 * Author: Andrew Retallack
 *        www.crash-bang.com
 */

#define F_CPU 16000000UL    //Clock running at 16MHz. Need to define this prior to including
                          // "delay.h"

#include <avr/io.h>        //Standard support for AVR I/O registers
#include <util/delay.h>    //Library to handle delays

int main(void)
{
    DDRB = DDRB | (1<<DDB0);    //Set PB0 as output - same as pinMode(x, OUTPUT)

    while(1)
    {
        //Turn LED on
        PORTB = PORTB | (1<<PORTB0);    //Set PB0 high - same as digitalWrite(x, HIGH)
        _delay_ms(1000);                //Delay for 1 second - same as delay(1000)

        //Turn LED off
        PORTB = PORTB & (~ (1<<PORTB0) );    //Set PB0 low - same as digitalWrite(x, LOW)
        _delay_ms(1000);                //Delay for 1 second - same as delay(1000)
    }
}

```

Listing 1: Source code for the project.

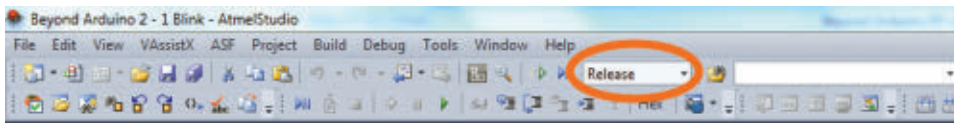


FIGURE 10: Choose Release from the menu bar.

Finally, you'll be presented with a window to enter your code into. For now, enter the code in **Listing 1** into the window (download it from the article link). We'll dissect it once we've got the LED flashing.

Step 5: Compile the Code

To compile the code, first make sure that you're working in a "Release" configuration, not a "Debug" configuration. The Debug configuration allows you to step through and debug your code in Atmel Studio before you've even uploaded it to your microcontroller. We want to compile it for the microcontroller, so choose "Release" from the menu bar (see **Figure 10**). To compile, hit the "F7" button on your keyboard. You should see a whole lot of activity in your Output window at the bottom of the screen; hopefully, you see a message that says:

```

Build succeeded.
===== Build: 1 succeeded or up-to-date,
0 failed, 0 skipped =====

```

Step 6: Flash the Code

Uploading the code is pretty straightforward if you're

using one of Atmel Studio's supported programmers or debuggers (see [7] and [8] in **References**). Simply connect the programmer to the USB port, click on the Debug menu, then "Start without Debugging."

If you're using a USBTiny or a USBASP, then you need to first configure Atmel Studio to use the programmer. This is needed once-off per microcontroller and is really quick:

1. Make sure you've downloaded the latest version of AVRdude (see [10] in **References**). AVRdude is a command-line program that allows you to upload code to AVR microcontrollers — more on AVRdude in a later article.

2. Under the Tools menu, click on "External Tools ..."
3. Click "Add" and enter the following:

- a. Title: Give this configuration a name (e.g., USBTiny ATmega328P).
- b. Command: This is the path to AVRdude.exe, including the "AVRdude.exe." On my PC, this reads "D:\AVRdude\AVRdude.exe."
- c. Arguments: These need to be typed in carefully. We'll go into detail at a later stage, but for now (assuming you're using an ATmega328P

Release\\$(ItemFileName).hex:!"

e. Check "Use Output Window."

Figure 11 shows the settings for a USBTiny external programmer.

Step 7: Watch It Blink

If you've found your way through all these steps, disconnect your programmer's USB cable from your PC and connect a power source. You'll now see the steady reassuring blink of your LED. Give a long contented sigh!

When you snap out of the LED-induced trance, move on to the next section to see how Atmel Studio understood the code.

Dissecting the Code

Let's wrap up by connecting the earlier discussion on registers and bitwise operations with the actual Atmel Studio code. I've added line numbers into the listing to make the lines easier to reference (refer to **Figure 12**).

Line 13: `F_CPU` defines the speed that the clock is operating in Hz. We've added six zeros for a value of 16 MHz, as well as a "UL" to specify that the type is an unsigned long. `F_CPU` is commonly used in libraries; for our purposes here, we need it for the `_delay_ms()` function (which resides in the `delay.h` library).

Line 15: This is included by default in new projects, and defines (among others) the “friendly” names for all the registers (ports and pins) we use here (remember registers are actually memory locations). These names are microcontroller-specific of course.

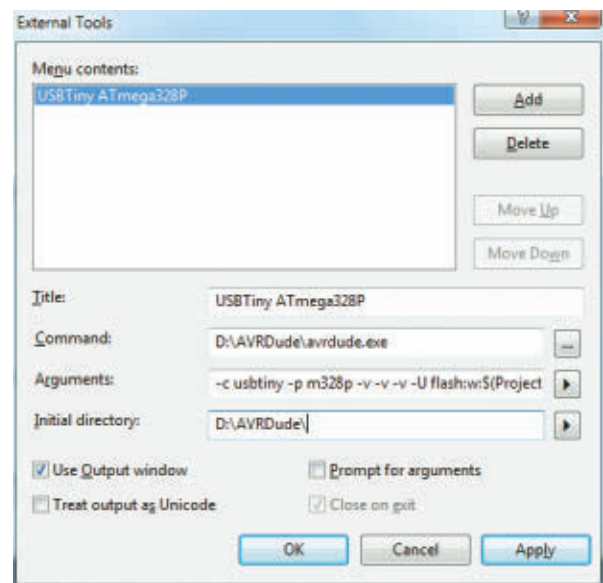


FIGURE 11: Configuring the USBTiny as an external programmer within Atmel Studio.

Line 16: This is a library of utilities to perform delays. We use the `_delay_ms()` function from this library later in the code.

Line 19: Wait! Where have the *setup()* and *loop()* structures gone? Well, there aren't such things outside of the Arduino IDE; just a *main()* function. Inside the main function, you need to perform all your setup steps before you enter an infinite *while(1)* loop. The *while(1)* is equivalent to the *loop()* structure in the Arduino.

Line 22: *DDRx* is a “direction register” for port x – it’s similar to the *pinMode()* function. So, *DDRB* sets the direction for each of the pins on port B. A 1 means an output and a 0 means an input. The *DDB0* refers to pin 0. Left-shifting (<<) a 1 by 0 positions actually leaves the 1 in its original position – i.e., 0b00000001.

By ORing this with the *DDRB* register, we are simply setting bit 0 to a 1. In other words, we're telling the *DDRB* register that pin PBO is an output. Note that even though the left shift here doesn't actually shift the bit to the left (as it's a zero), it's good practice to use this syntax so as to be unambiguous.

Line 24: This *while(1)* infinite loop is the same as the *loop()* function in the Arduino IDE.

Line 27: *PORT_X* is the



FIGURE 12: Dissecting the source code.

register containing the bits that control the high/low state of the pins on port x — a lower-level version of `digitalWrite()`. Setting bit 0 in PORTB controls the high/low state of pin PB0; “1” drives the pin high, and “0” drives the pin low. The PORTB0 is a macro that refers to pin 0. In the same way as we set the direction register `DDRB`, we set bit 0 of the PORTB register to a 1 using the boolean OR operator.

Line 28: Calls a function from the `delay.h` library,

causing the microcontroller to wait for one second. This is equivalent to the `delay()` function in the Arduino IDE.

Line 31: Here, we clear bit 0 in the PORTB register, setting pin PB0 low.

Line 32: Another delay for a second, before we start all over again.

What's Your Name?

Hopefully, the above code makes sense. If you were to start a program from scratch, though, how on earth would you know the names of the ports, pins, direction registers, and more? The answer lies in the confusing, often unfathomable datasheet. They're called datasheets and not information sheets for a good reason. It takes some skill and experience to find what you need in the 500+ pages.

For information on the I/O ports and pins, the place to go is section 14 of the datasheet (for the ATmega328 at least). The overview section covers the naming of the registers, and then goes into detail on how to use them for general input/output functions. I initially found the language difficult to understand, but eventually I started to see patterns and worked out what the authors were trying to get across.

The register and pin names from the datasheet are defined in the “`io.h`” file that is included by default in all new projects. This is what enables you to refer to them by their datasheet names.

What's Next?

Whew! We've covered a lot of ground in this second article — hopefully enough to get you working on some interesting projects over the course of the next month. We've covered the fundamentals of creating simple digital outputs, but I'm sure you're wondering how to create programs that accept input.

Next month, we'll dig into handling inputs from users in order to create a more interactive project.

NV

ALL ELECTRONICS

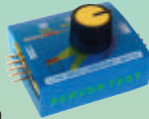
www.allelectronics.com
Order Toll Free 1-800-826-5432

SERVO TESTER

Ideal for RC hobbyists. Connect up to three servos simultaneously to test and compare speed and reaction time without use of transmitter/receiver. Power input 4.8-6 Vdc. 1.3" x 1.1" x 0.5" high.

CAT# STR-110

\$10.00
each



STANDARD RACING SERVO

Standard size analog servo motor. 49oz-in torque @ 6Vdc. 0.19 sec/60° @ 6V. 6.5" leads w/ JR connector.

CAT# DCS-110

\$6.95
each

5 for \$6.75 each



SOLDERLESS BREADBOARD, 400 CONTACTS

Accommodates all sizes of dips and discrete components. Interconnect with solid hook-up wire. Wires and components can be re-used many times without damage to board or components.

CAT# PB-400

\$4.00
each



3 1/2 DIGIT LCD MULTIMETER W/ BACKLIGHT

Velleman # DVM850BL. For features, accuracy and ruggedness, this is the best inexpensive multimeter we've ever seen. DC current (10 A), DC and AC voltage (600 V), resistance (2 M ohm), diode, transistor tester, audible continuity and hold button. Protective rubber shell and test leads included.

CAT# DVM-850BL

\$17.95
each



24 VDC GEAR MOTOR W/ TURNTABLE

Low-current 24 Vdc motor and gearbox with a 4.9" diameter turntable. Turntable can be removed from motor. 45RPM @ 24 Vdc, 60mA - operates at 12Vdc at 1/2 speed. Drive motor, 1.1" dia. x 1.67". Gearbox, 2.75" x 3.00" x 0.57". Threaded mounting holes in four corners. 5/16" dia. shaft, flatted & splined.

CAT# DCM-351

\$9.75
each



SET OF 5 GEARS AND BUSHINGS

Five matched break-resistant plastic gears ranging from 10 to 50 teeth, 11-51mm diameter. Gears have 4mm bores and include inserts which reduce the bore to 2mm diameter.

CAT# GR-5

\$2.75
each

10 for \$2.00 ea.



3-PHASE BRUSHLESS DC MOTOR

BEI # DIH23-30-0132. Smooth, quiet, high-torque brushless DC motor. 2.25" diameter x 3.0" long. 0.25" diameter x 1" long shaft. 5270 RPM with no load. Built in speed sensor. 7-15Vdc operating voltage.

Note: This type of DC motor requires a controller to function. We have some inexpensive controllers and servo testers that can be used to operate this motor. See our website for more information.

CAT# DCM-459



\$16.95
each

NEW PRODUCTS *Continued from page 23*

contactor is 1.5 amps at 20°C temperature rise. Socket temperature range is -55°C to +180°C. The socket also features an alignment guide for precise device-to-pin alignment. The specific configuration of the package to be tested in the CBT-QFN-7039 is a QFN, 3x3 mm, 0.5 mm pitch, and 10 positions with a center ground pad. The socket is mounted using supplied hardware on the target printed circuit board (PCB) with no soldering.

To use, place the QFN device into the socket base and lock the double latch socket lid onto the base using the latch. The socket uses a compression wave spring to apply constant downward pressure, enabling the device to be interconnected to the target PCB. This socket can be used for hand test and quick device screening applications with the most stringent requirements.

Pricing for the CBT-QFN-7039 is \$552 each, with reduced pricing available depending on quantity required.

For more information, contact:
Ironwood Electronics
Web: www.ironwoodelectronics.com

EDGE-LIT LED GLASS FLAT PANEL LIGHTING

Super Bright LEDs introduces their new decorative edge-lit LED glass flat panel lighting, which lends itself to new construction applications or for retrofitting existing fluorescent light fixtures. It's available in round and square versions with edge-lit glass bezel.



Decorative panels can provide both accent, as well as task lighting. Their super-thin flat panel design installs in tight spaces such as finished basement ceilings and around duct work.

These recessed lights have an integral heatsink for cooler running temperatures and a UL-recognized constant current driver for enhanced reliability. They are available in a natural white or warm white color temperature with white housing. They have a 120 degree beam angle.

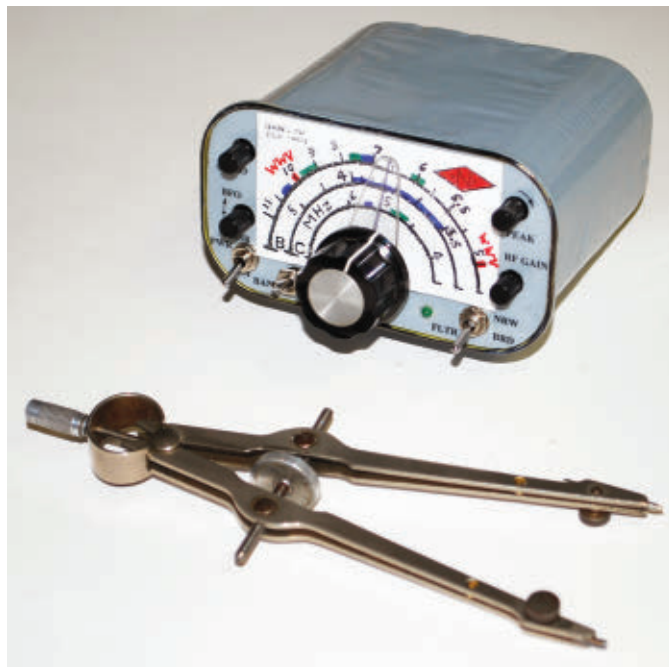
With an average current draw of 300 mA, these panel

light fixtures can provide lighting needed at a fraction of the power cost. Various sizes to choose from include 2 x 4, 2 x 2, and 1 x 2 feet.

For more information, contact:
Super Bright LEDs
Web: www.superbrightleds.com

COMPACT HIGH FREQUENCY RECEIVER

National RF, Inc., announces the first of several new high frequency and shortwave receivers to be made available in the near future. Designed specifically for the electronic enthusiast and shortwave listener who desires a very unique but highly functional and highly portable shortwave radio, National RF is now offering the 75-NS-3 receiver. The receiver is available as a semi-kit in which the main circuit board is loaded and functionally tested at the National RF facility. The customer is responsible for procuring the enclosure, and must do the drilling and integration of the enclosure to the electronic assembly.

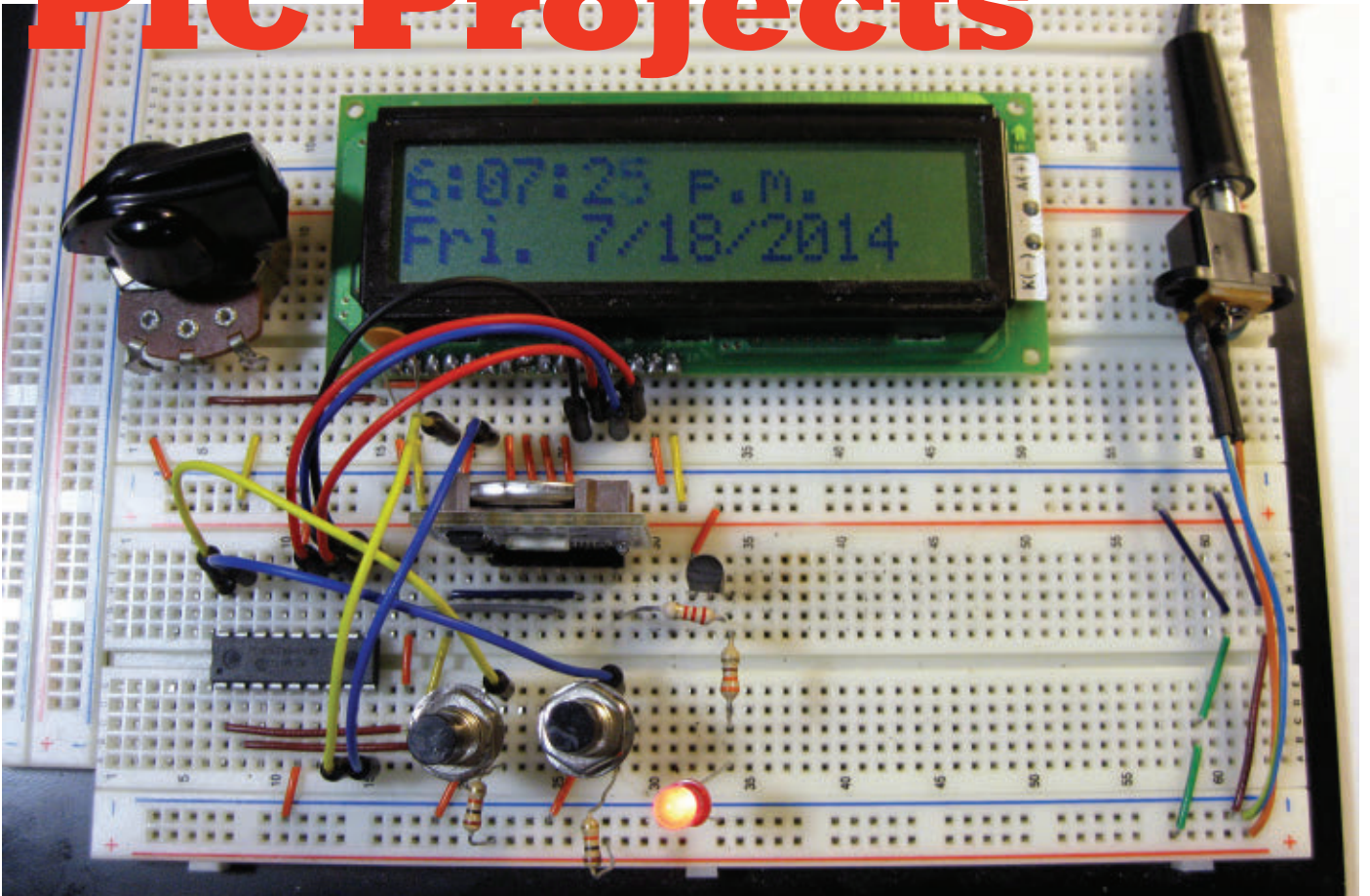


The receiver is designed to cover the 3.5 MHz to 10.7 MHz frequency range, which includes the 80-meter, 60-meter, 40-meter, and 30-meter amateur bands, as well as several international shortwave bands, plus WWV time and frequency standards at 5 and 10 MHz. The procured enclosure is a potted meat can (whose copyrighted name is synonymous with unwanted emails).

Continued on page 77

Add a Real Time Clock to Your PIC Projects

By Thomas Henry



Real time clocks have all sorts of neat uses in microcontroller projects. Most obvious would be in applications such as alarm clocks, desk calendars, telescope controllers, smart thermostats, automatic lawn sprinklers, and so forth. They are also useful for creating timestamps in recordings, logged data from weather stations, hourly energy usage reports, and much more.

While it's certainly possible to roll your own microcontroller clock with the addition of little more than a clock crystal and some assembly language, in this golden age of inexpensive electronics for DIY, it makes more sense to go with a dedicated add-on device. This article introduces you to the world of timekeeping with just such a module: the popular and inexpensive TinyRTC.

The TinyRTC unit is a complete clock/calendar set to shoulder all of the timekeeping duties within your circuits.

It's readily available from a wide range of suppliers for around six bucks.

You're going to love the TinyRTC's features:

- Clock/calendar
- 4096 bytes of EEPROM
- 56 bytes of user RAM
- Rechargeable backup battery
- Communicates effortlessly along the I²C bus
- Dedicated pulse output for interrupt applications

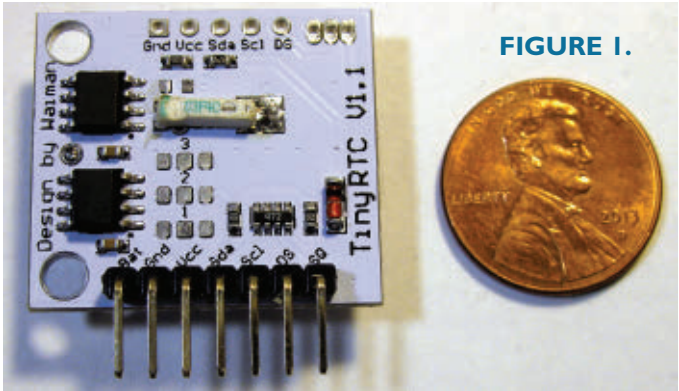


FIGURE 1.

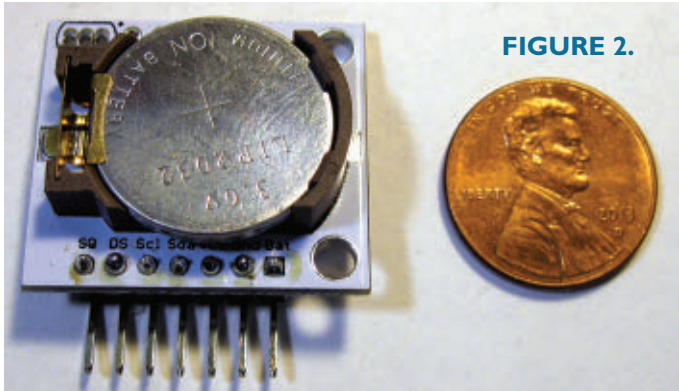


FIGURE 2.

When you get right down to it, the TinyRTC is really little more than a breakout board for the DS1307 clock chip and the AT24C32 EEPROM. So, everything described in this article can be applied to those two ICs should you really want to tax your eyes and solder up some surface-mount chips on your own. You can spot the two tiny integrated circuits in **Figure 1**.

The reverse side (shown in **Figure 2**) shows the rechargeable lithium-ion battery in its piggyback holder. If it isn't obvious, this means that the clock continues to keep time even if main power to the circuit is absent. When power is there, the battery is kept fully charged.

It's pretty clear that the module was originally created with the Arduino crowd in mind. In fact, that community already has libraries available for putting the TinyRTC through its paces. Many people — me included — prefer working from scratch with the PIC for customized minimalist implementations costing far less than the Arduino approach, however. Does that mean we have to start all over again in assembly language? Not at all! I'll show you how to easily get the TinyRTC up and running on a PIC and — best of all — in the easy-to-use Basic language. Let's tuck in.

All About the Clock Registers

As mentioned, at the heart of the TinyRTC is the well-known DS1307 chip. Within this IC are eight registers which allow you to set things up and monitor the time and date. **Figure 3** gives the details. Though pretty self-explanatory and complete, there are a few points deserving a little extra explanation for newcomers.

The various numbers stored within these registers are in binary-code-decimal (BCD) format, which is

particularly handy when printing things out to an LCD, among other things. Recall that in this scheme, a single byte (eight bits) can store a two digit number; the lower nibble specifies the unit's place, and the higher nibble gives the ten's place. As it turns out, when referring to time and date, not all of the bits within the higher nibble will be needed.

For example, with minutes and seconds, the greatest number which can occur is 59, meaning that only three bits are required to represent the ten's place digit.

What about dates? The largest number you'll have to worry about is 31 — the greatest number of days to appear in a month. In this case, only two bits need to be spent in the higher nibble to represent that ten's place

	7	6	5	4	3	2	1	0								
Register 0	Halt		Seconds: Ten's			Seconds: Units										
Register 1	0		Minutes: Tens			Minutes: Units										
Register 2	0		0		Hours: Tens		Hours: Units									
	0		1		0 = am 1 = pm		Hours: Tens		Hours: Units							
Register 3	0		0		0		0		0		Day of Week					
Register 4	0		0		Date: Tens			Date: Units								
Register 5	0		0		0		Month: Tens		Month: Units							
Register 6	Year: Tens					Year: Units										
Register 7	Out		0		0		SQWE		0		0		RS1		RS0	

When SQWE is high, RS0 and RS1 control the Square Wave output, according to the scheme at the right. Otherwise, when SQWE is low, the output simply follows the value of Out (bit 7).

RS1	RS0	OUTPUT
0	0	1 Hz
0	1	4096 Hz
1	0	8192 Hz
1	1	32768 Hz

FIGURE 3.

digit. The TinyRTC takes advantage of any bits not needed for other purposes, or sometimes just locks them to zero. For example, bit 7 of Register 0 is used to turn the clock on or off. If high, then the clock is halted, but if cleared it takes off running.

Register 2 can be used in two different ways as shown in the **figure**. If its bit 6 is low, then time is represented in 24 hour mode (so-called military time). On the other hand, if bit 6 is set, then time is stored in a 12 hour format. In the latter case, bit 5 tells you if the hour is before or after noon; it's cleared for the former, or set for the latter.

The other time and date registers should be pretty obvious, so let's turn to Register 7 which controls the output pin labeled SQW, standing for "square wave." This pin is brought out on the TinyRTC and can be used in a couple interesting ways. As the **figure** indicates, if bit 4 is set, then a square wave is generated on the SQW pin. The bits labeled RS0 and RS1 let you set the frequency at 1 Hz, 4.096 kHz, 8.192 kHz, or 32.768 kHz.

The 1 Hz option is particularly attractive since it could be employed to generate microcontroller interrupts once a second; say, to take a new temperature reading, or just to flash the colon between the hours and minutes of a display. If bit 4 is cleared, then the output pin simply follows the state of bit 7 which gives you software control of SQW. Take a moment to look over the **figure** one last time. While it might appear mysterious at first blush, you'll find yourself at home with it quite soon.

Bring On the Memory

As mentioned earlier, the TinyRTC gives you access to 56 bytes of general-purpose RAM within the DS1307 chip; you can use this any way you want. The addresses

Command	Purpose
<i>RTC_Enable()</i>	Turn clock on or off
<i>RTC_ResetClock()</i>	Reset clock to manufacturer's default
<i>RTC_SetClock()</i>	Set date, day of the week, time, and hour mode
<i>RTC_SetTime()</i>	Set only the time
<i>RTC_SetDate()</i>	Set only the date and day of the week
<i>RTC_ReadClock()</i>	Read date, time, and am/pm flag
<i>RTC_ReadTime()</i>	Read only the time and am/pm flag
<i>RTC_ReadDate()</i>	Read only the date and day of the week
<i>RTC_SetHourMode()</i>	Choose 12 or 24 hour clock mode
<i>RTC_SetSQW()</i>	Set SQW output frequency
<i>RTC_Write()</i>	Write to clock register or RAM
<i>RTC_Read()</i>	Read from clock register or RAM
<i>eepByteW()</i>	Write a byte to the EEPROM
<i>eepByteR()</i>	Read a byte from the EEPROM
<i>eepArrW()</i>	Write a string or array to the EEPROM
<i>eepArrR()</i>	Read a string or array from the EEPROM

Figure 4.

follow immediately after the registers described previously. So, RAM lives at locations 8 through 63.

We normally think of RAM as temporary memory. Thanks to the backup battery, however, it now becomes nonvolatile — at least as long as you keep the cell in place.

The AT24C32 IC provides 4096 bytes of genuine nonvolatile EEPROM storage. This might prove useful for such things as storing messages to be displayed on an LCD at times directed by the clock (birthdays, appointments, warnings, etc.) or perhaps for logging data from outboard sensors (light, temperature, humidity, and so forth). In any event, the RAM and EEPROM are there for the taking, so feel free to drink deeply of them.

Start Commanding the Module

Both the DS1307 and AT24C32 chips on the TinyRTC module are I²C devices. If you've played with this synchronous communication scheme before, then you'll know that the two lines called SCL and SDA (clock and data, respectively) must have pull-up resistors on them. The TinyRTC includes these. Even though both chips are hanging on the SCL and SDA lines, only one set of pull-ups is required.

I hope I haven't frightened you with the mention of I²C — a scheme that seems daunting at first. For, in fact, the software library put together especially for this article takes all of the wailing and gnashing of teeth out of it! A handful of higher level commands is all you need to start doing something useful at once. Let's see what they look like. The software to drive the TinyRTC has been written in Great Cow Basic. This is an amazingly full-featured, powerful, yet easy-to-use compiler. Best of all, it's open source and absolutely free of charge. You can download it from gcbasic.sourceforge.net. You can fetch this article's source code from the article link.

Apart from the four demonstration programs that we'll get to in just a moment, there are two other files of note. These are general-purpose *include* files which add I²C and TinyRTC commands to the Great Cow Basic compiler. They're titled "I2C-Alt.h" and "TinyRTC.h," respectively. (Be sure to view the "ReadMeFirst.txt" file for any updates). Actually, the I²C stuff will be invisible to you, and is required only by the TinyRTC file. Of course, you're certainly welcome to dip into whatever's there should you wish to modify the TinyRTC commands or create new ones which occur to you. That's the beauty of open source software.

So, what are these high level commands? **Figure 4** lists them. These should all make a great deal of sense. There are commands to set the time, read the date, set the SQW output frequency, read and write to the EEPROM, and so forth. To cut down on confusion in the **figure**, the parameters aren't shown. Instead, if you head

to the *include* source file “TinyRTC.h,” you will find a very thorough description of what each command does and what parameters are required. The code itself is completely commented from top to bottom should you wish to learn how it all works.

Incidentally, Great Cow Basic has a very rich set of string commands, and it’s a snap to read or write not only bytes to EEPROM, but also entire strings or arrays. Click!

Time for Some Experiments

Enough of this palaver. Let’s head straight to the lab and start doing something on the breadboard. I’ve put together four experiments you can try right away. **Figure 5** shows the schematic for everything. The source files for each experiment are also at the article link. As usual, the code is heavily documented, so take time to read over the comments to learn even more about how this remarkable module can be harnessed.

Back to the **schematic**. Apart from the TinyRTC, there’s absolutely nothing exotic appearing here. In fact, you probably have all of the required parts in your lab right now; everything is completely generic. The entire affair can be breadboarded in under 15 minutes, yet reveals just about every aspect of the TinyRTC. Before starting, do take the usual precaution of double-checking the power connections (+5V and ground) to both the clock module, as well as the PIC.

Incidentally, I used the common and inexpensive PIC16F88, but Great Cow Basic makes it a snap to change over to some other PIC if you prefer. The I²C routines at the heart of this are software in nature (“bit-banging”) and so will work on most any pin of most any PIC.

In Experiment #1, you’ll learn how to read from and write to the RAM one byte at a time. The LCD actually displays the instructions, so go ahead; run it and see for yourself. Experiment #2 is similar, but now communicates with the EEPROM; again, one byte at a time. In Experiment #3, you’ll write an entire string to EEPROM in one fell swoop.

Finally, Experiment #4 is the one you’ve been waiting for: a complete clock project. You can set the date and time, and it takes off in perpetuity, with the LED blinking

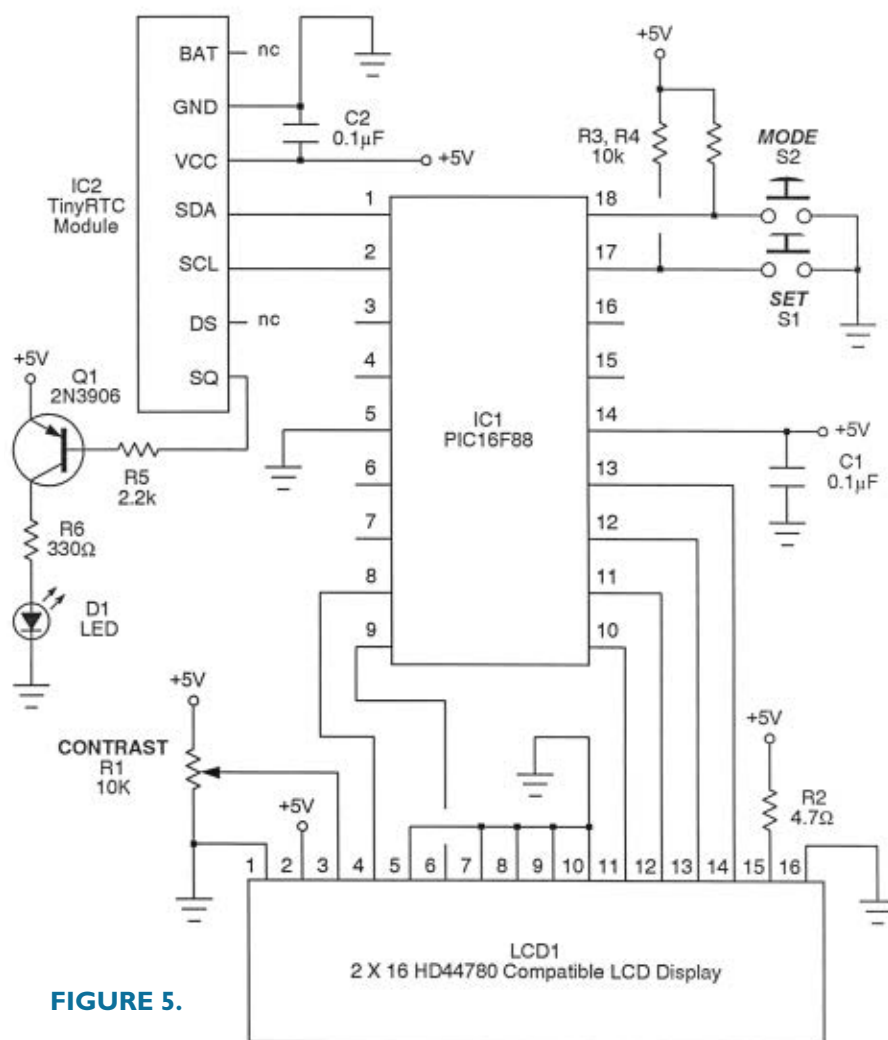


FIGURE 5.

A 0.1 μF decoupling capacitor should also be used on LCD1 (not shown). Choose R2 to match the backlight requirements of the LCD used.

once per second. Even if you disassemble the breadboard and set the TinyRTC back up on the shelf, it still keeps time thanks to the lithium-ion battery.

One last thing before turning you loose to design your own custom applications. It’s only natural to wonder about the accuracy in a time-keeping device. The DS1307 within the TinyRTC is reasonably well-suited to many common applications. I find mine loses about three seconds a day. Naturally, this changes according to temperature and who knows what else — even under crystal control. If this concerns you overly, then you’ll be interested to know there is a TinyRTC clone featuring the DS3231, which contains an integrated temperature compensated crystal. (The EEPROM remains the same). While considerably more accurate, it’s still backwards compatible with the TinyRTC *include* files and programs of this article.

So, get tick-tocking on your PIC today ...

NV

Driving LEDs with a Microcontroller

By Craig A. Lindley

Post comments on this article and find any associated files and/or downloads at www.nutsvolts.com/index.php?/magazine/article/april2015_Lindley.

One of the first experiments people learning about microcontrollers usually perform is how to control an LED. Typically, they hook up an LED in series with a current-limiting resistor, connect it to an output pin, and write some simple software to make it blink. The Arduino blink sketch shown below is an example:

```
int led = 13;

// the setup routine runs once before the loop() function
void setup() {
    // initialize the digital pin as an output.
    pinMode(led, OUTPUT);
}

// the loop routine runs over and over again forever:
void loop() {
    digitalWrite(led, HIGH);      // turn the LED on (HIGH is the voltage level)
    delay(1000);                 // wait for a second
    digitalWrite(led, LOW);      // turn the LED off by making the voltage LOW
    delay(1000);                 // wait for a second
}
```

On Arduinos, there is an LED and current-limiting resistor on board already (and connected to pin 13), so there is nothing really to hook up for this first experiment. Once this sketch is downloaded onto your Arduino, you should see the onboard LED blink on and off until power is removed.

As you might expect, the thrill of watching the LED blink wears off pretty quickly. Next, people might want to try and control the brightness of an LED with software.

The following Arduino fade sketch causes the LED connected to pin 9 through, say, a 470 ohm resistor to ground, to go from off to full brightness and then back down, over and over:

```

int led = 9;           // the pin that the LED
is attached to
int brightness = 0;    // how bright the LED is
int fadeAmount = 5;    // how many points to
fade the LED by

// the setup routine runs once before the loop()
function
void setup() {
  // declare pin 9 to be an output:
  pinMode(led, OUTPUT);
}

// the loop routine runs over and over again
forever:
void loop() {
  // set the brightness of pin 9:
  analogWrite(led, brightness);

  // change the brightness for next time through
the loop:
  brightness = brightness + fadeAmount;

  // reverse the direction of the fading at the
ends of the fade:
  if (brightness == 0 || brightness == 255) {
    fadeAmount = -fadeAmount ;
  }
  // wait for 30 milliseconds to see the dimming
effect
  delay(30);
}

```

In this sketch, an *analogWrite* statement is used to control LED brightness instead of the *digitalWrite* command used to turn the LED off and on as in the previous sketch. Brightness control works using a combination of persistence of vision coupled with Pulse Width Modulation, or PWM for short.

Persistence of vision is an effect where our eyes and brain hold onto an image we see for approximately 1/25th of a second before it fades away. We all experience this effect at the movies where we fail to notice that a motion picture screen is actually dark about half the time. Motion pictures project one new frame every 1/24th of a second. Each frame is shown three times during this period. Our eyes retain the image of each frame long enough to give us the illusion of smooth motion. How does this relate to LED brightness? Glad you asked.

An LED — being a semiconductor device — can be switched on and off very quickly. An LED is at full brightness when it is on all of the time over a fixed period of time. If the LED is only on half of the same time period (and off the other half of the time period) and the time period is very short, it will appear approximately half as bright. Now, if this switching happens at a fast enough rate, our persistence of vision will not perceive the LED as being turned on and off or flickering, but will perceive it as being on at some brightness level.

PWM divides up the periodic time period into intervals based on the resolution of the hardware. On most eight-bit microcontrollers, eight-bit PWM is supported. This means that there are 256 unique durations

from always off to always on. Duty cycle is defined as the ratio of on to off times. A PWM output that is on half of the time has a 50% duty cycle.

Figure 1 illustrates various duty cycles of a PWM output. The green lines in this figure show the periodic nature of the PWM output. If a PWM output is used to control LED brightness, the frequency of the PWM output becomes important. Flickering of the LED will be visible if the PWM frequency is too low. Most — if not all — microcontrollers allow the frequency of their PWM outputs to be configured.

The *analogWrite* function in the previous sketch sets how long the PWM output connected to the LED is on; *analogWrite(0)* means the output is never on and the LED is dark; *analogWrite(255)* means the PWM output is always on, so the LED is at full brightness. Values between 0 and 255 determine the relative brightness of the connected LED.

We should quickly say a few words about current limiting with LEDs. Current limiting is important to protect both the digital output of the controller driving the LED and the LED itself from burning out. To figure out the value of a current-limiting resistor to use with an LED requires three pieces of information. First, the supply voltage used to drive the LED (V_s); second, the current (I) you want to operate your LED at; and third, the forward voltage (V_f) drop of the LED. Forward voltage varies by the color of the LED. A red LED typically drops 1.8 volts whereas a blue LED may drop 3.3 volts.

As an example, assume our supply voltage is five

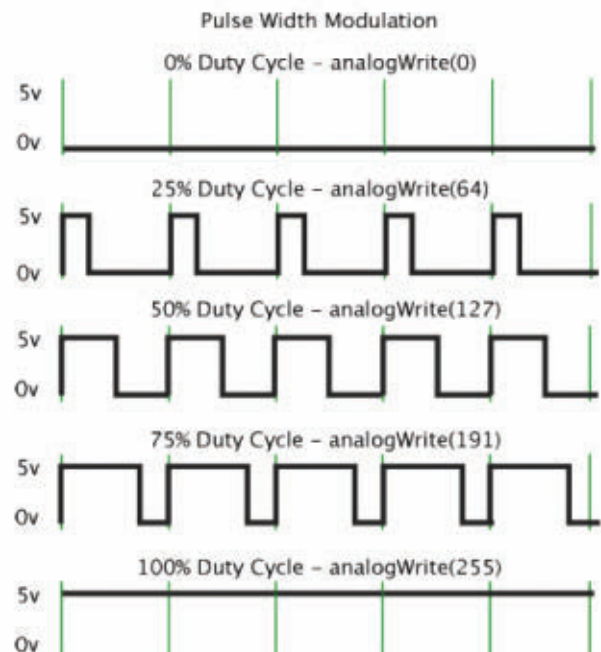


FIGURE 1. Pulse width modulation and duty cycles.

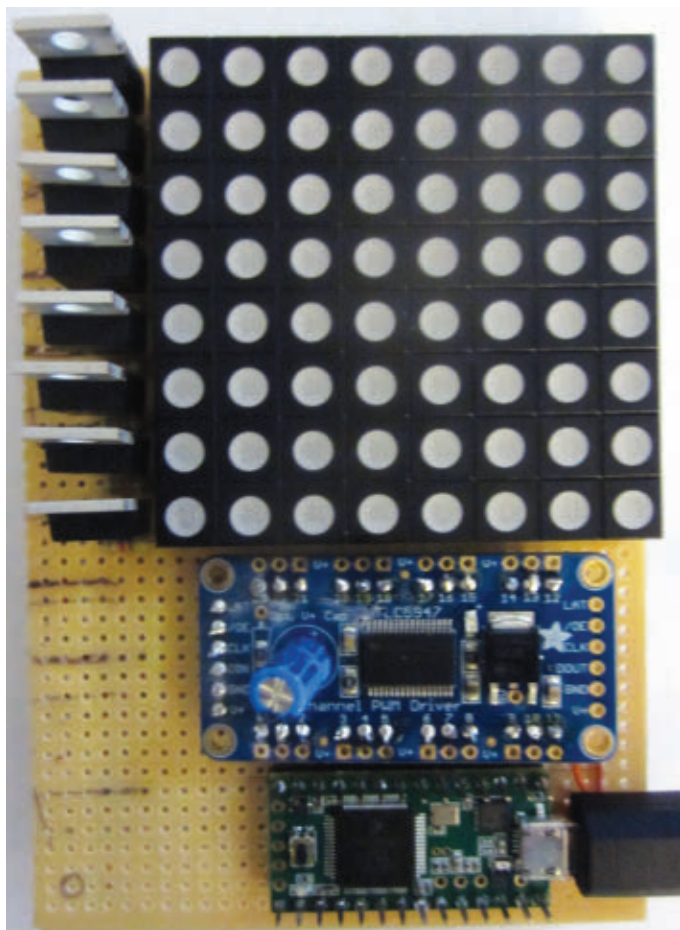


FIGURE 2. The demonstration hardware.

volts; assume we want 20 mA (0.02 amps) of current for the LED, and the voltage drop across the LED is 1.8 volts. Using Ohm's Law, we can calculate the required resistance with the formula:

$$R = (V_s - V_f) / I$$

which works out to be around 160 ohms. If the resistor value you calculate turns out not to be a standard value, pick the next larger value to be safe.

Okay, so now we see how the brightness of an LED

can be controlled using PWM. With this information, you could control the brightness of a red, green, blue, or any single color LED with software. What if you want variable color output from an LED? In this case, you would probably choose an RGB LED for this purpose. RGB LEDs actually contain red, green, and blue LEDs internally. This means three PWM channels would need to be used to control the brightness and the color of a single RGB LED. By varying the duty cycle of each of the LEDs inside the RGB LED, many color combinations are possible. If eight-bit PWM is used on all three channels driving an RGB LED, there are theoretically 256 x 256 x 256, or over 16 million possible color combinations. Research has shown the human eye can discern approximately seven million unique colors.

On most Arduino boards (those with the ATmega168 or ATmega328), PWM is available on pins 3, 5, 6, 9, 10, and 11. On the Arduino Mega, it works on pins 2-13 and 44-46. Older Arduino boards with an ATmega8 only support PWM on pins 9, 10, and 11. On the Teensy 3.1 microcontroller that I typically use, PWM is available on pins 3, 4, 5, 6, 9, 10, 20, 21, 22, 23, 25, and 32.

So, as you can see, on typical microcontrollers there are only a small number of PWM outputs available for driving LEDs. If you only want to drive single color LEDs, you may be okay, but if you want to drive a large number of RGB LEDs the outlook is bleak.

There are many options available for driving larger numbers of LEDs using hardware external to but controlled by a microcontroller. If you search the Internet, you will see many examples. Many designs use 74HC595 shift register chips to drive the LEDs, but my current solution of choice is the Adafruit 24-channel 12-bit PWM LED driver with SPI interface (product #1429; available for \$14.95). With its 24 channels, you can control 24 single color LEDs or eight RGB LEDs with the added advantage of 12-bit PWM, giving finer grain control than the eight-bit PWM described previously.

An additional advantage is that each PWM output provides constant current, so current-limiting resistors are unnecessary. In fact, one resistor on this driver board controls the current through all channels which is set at the factory to 15 mA. Up to 30 mA of drive current per channel is possible by changing the resistor. NOTE: This device is a current sink. It sinks current to ground; it cannot source current.

The Adafruit device is really just a breakout board for the TLC5947 controller chip from TI. This breakout board makes all of the TLC5947 signals available in an easy-to-use configuration without having to deal with surface-mount components.

The TLC5947 chip is a cascadable shift register with built-in PWM counters and PWM

Qty	Part	Description	Source
1	Display 1	Common anode 8x8 RGB LED matrix	eBay
1	U2	24-channel PWM controller	Adafruit
1	U1	Teensy 3.1 controller	pjrc.com
8	R1-R8	4.7K 1/4 watt resistor	Radio Shack
8	Q1-Q8	P channel power MOSFETs	SparkFun
1		Breadboard	RadioShack
1		USB cable with five-pin micro-B plug for connecting Teensy to a computer or USB power supply	pjrc.com
1		Optional USB power supply capable of at least 1A @ 5V	RadioShack

oscillator. Since each output requires 12 bits of information to control its PWM hardware, a total of 288 bits or 36 bytes of data must be streamed into the chip via SPI to control its 24 PWM outputs. (More on how to control the TLC5947 chip in the software section).

For many of my projects, a single Adafruit driver board still doesn't have enough outputs to drive large numbers of RGB LEDs. I recently purchased some 8x8 RGB LED matrices for use in a project. Think about it. This is a matrix of 64 RGB LEDs which equates to 64 x 3, or 192 individual LEDs needing PWM control.

There were two ways to go about this project. Purchase and connect eight of these boards together somehow so each board controls one row of the display for a total driver cost of \$119.60. Or, use one of these boards, eight cheap P channel power MOSFETs and some clever software to control the entire display.

Being frugal, I chose the latter. In the discussion to follow, I will show you how to use multiplexing of the LED driver to accomplish this feat.

Multiplexing

Multiplexing is a term from the telecom industry which meant to combine multiple channels of data onto a single medium for transmission. Multiplexing reduces the cost of hardware, and because of a reduced parts count increases reliability.

Multiplexing many channels of data onto a single medium required that each channel of data be given its own time slot. This is referred to as TDM, or Time Domain Multiplexing. We can use this same technique for

controlling our LED matrix by assigning each row of the display a different time slot for update. If we update each row of RGB LEDs fast enough, persistence of vision will make it appear that each LED is individually controlled.

I designed some hardware to demonstrate control of an 8x8 RGB LED matrix using multiplexing. The hardware is shown in **Figure 2** and the hardware's schematic is shown in **Figure 3**.

The Demonstration Hardware

The 8x8 RGB LED matrix I will control is of the common anode variety. What this means is that each row of the display has the anodes of each red, green, and blue LED connected together (see **Figure 4**). The cathodes of each column of the same color LEDs are also connected together and brought out to pins on the matrix. By applying a current source to a specific row pin and a current sink to a specific cathode pin, a single color LED can be illuminated.

I tested the LED matrix I built into the demonstration hardware using a nine volt battery and a 1K ohm resistor by connecting the + side of the battery through the resistor to a row pin, and connecting the - side of the battery to the various column pins. As you move the battery connection from one column pin to another, you will see the LEDs change color.

As mentioned, in the demonstration hardware, P channel MOSFETs are used as the current source for each row of the matrix. The current flowing from the drain of the device into the row of LEDs is controlled by the

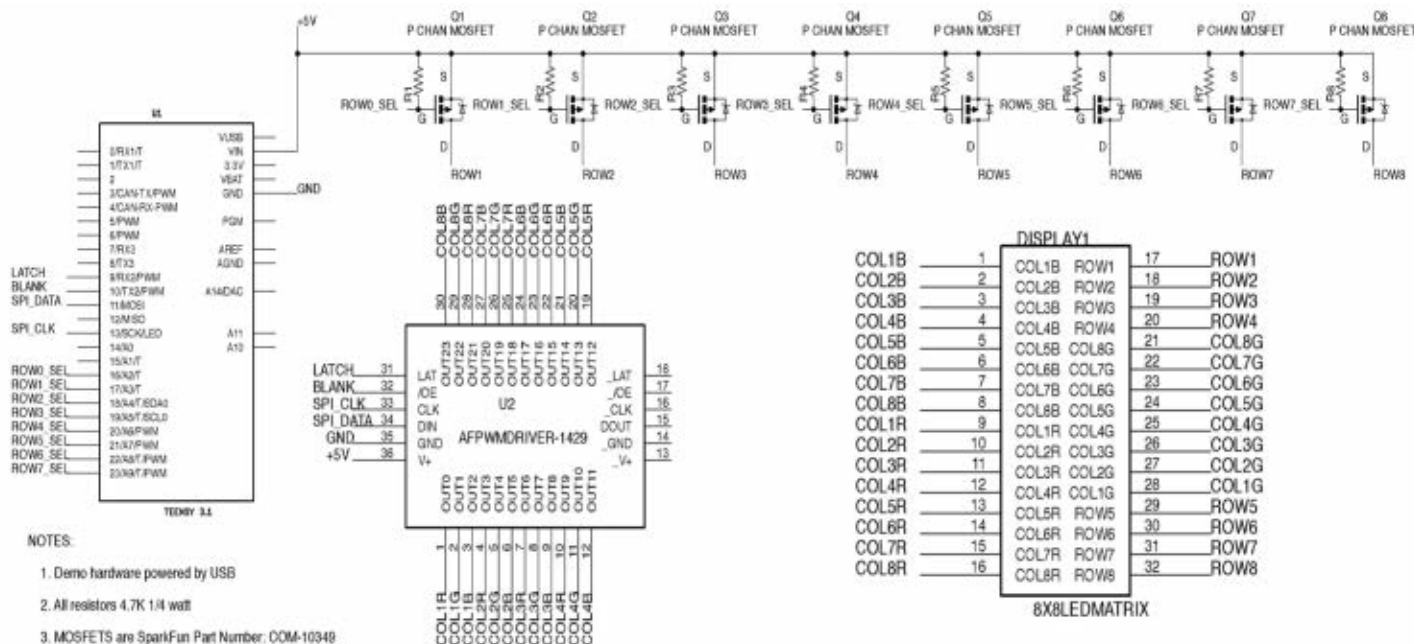


FIGURE 3. Demonstration hardware schematic.

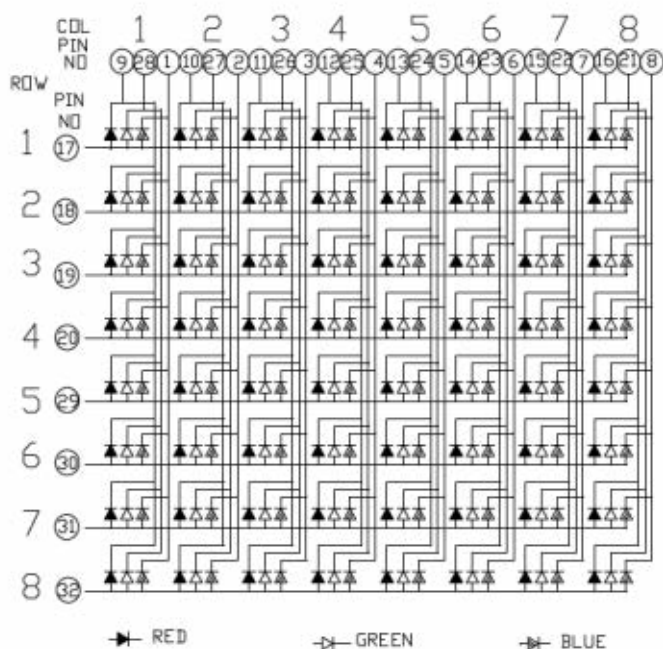


FIGURE 4. Common anode RGB LED matrix schematic.

voltage applied to the gate of the MOSFET. As wired, current flows to the LEDs in the row if the gate is driven low by the Teensy 3.1 controller. If the gate is pulled up to the voltage at the MOSFET's source pin, current flow stops. Each row MOSFET is connected to a different pin on the Teensy so that each can be individually controlled.

The current sink side of the equation is handled by the TLC5947 chip on the Adafruit board. The 24 PWM channels provided by the chip are connected to the 24 color column pins of the LED matrix. The data used to control the PWM outputs is streamed serially into the TLC5947 chip via the SPI interface on the Teensy controller. A data and a clock line are used to move the data which flows in one direction only.

Two other signals are needed for control of the TLC5947 chip. The first is labeled /OE on the Adafruit board, but called BLANK on the chip itself. When BLANK is high, all of the PWM channels stop sinking current and the chip's internal PWM counters are reset to zero. When BLANK goes back low, the current sinks are enabled and the PWM counters start counting.

The second control signal is labeled LAT on the Adafruit board but called XLAT on the chip. The rising edge of this signal causes the data contained in the TLC5947's shift register (loaded via SPI) to be transferred to the individual PWM counters backing each output pin.

I used a Teensy 3.1 controller from pjrc.com for controlling the hardware. It has plenty of RAM and Flash memory for coding up any kind of display patterns you can envision. The Teensy is Arduino compatible via the

Teensyduino software available at the website.

The hardware works as follows (under control of the software which will be described next): Data for a row of LEDs is moved from the controller to the TLC5947 chip using SPI. The data is latched into the chip on the rising edge of XLAT, and then the appropriate row select output is brought low to enable the LEDs in the row. Next, the BLANK line is driven low which causes the PWM counters to start and the LEDs in the selected row to illuminate. After a precise period of time, data for the next row is loaded and the whole process repeats indefinitely.

The Demonstration Software

The software running on the Teensy 3.1 controller is what makes multiplexing of the LED matrix possible. Multiplexing makes the hardware simpler, but the software more complex. The demonstration software sketch is available at the article link if you would like to duplicate what I have done or use pieces of the software in projects of your own. The sketch requires the TimerOne and the spi4teensy3 libraries to be available in your Arduino build environment. Please refer to the LEDMatrix8x8.ino sketch for the discussion that follows.

The first order of business in the sketch is to define the control signals/pins of the Teensy controller which will control the TLC5947 chip. The assignments are as follows:

```
// PWM driver control pins
#define LATCH_PIN 9
#define BLANK_PIN 10

// LED Matrix row select pins
#define ROW0_PIN 16
#define ROW1_PIN 17
#define ROW2_PIN 18
#define ROW3_PIN 19
#define ROW4_PIN 20
#define ROW5_PIN 21
#define ROW6_PIN 22
#define ROW7_PIN 23
```

An interrupt on the Teensy 3.1 controller is used to generate the precise timing required to make multiplexing work. The three values below define interrupt timing:

```
// Interrupt period calculations
#define DATA_XFER_TIME_USEC 13
#define ROW_DISPLAY_TIME_USEC 1024

#define INTERRUPT_TIME_USEC (DATA_XFER_TIME_USEC
+ ROW_DISPLAY_TIME_USEC)
```

DATA_XFER_TIME_USEC is the time in microseconds it takes for the spi4teensy3 library to transfer 36 bytes of row data from the Teensy running at 96 MHz to the TLC5947. The *ROW_DISPLAY_TIME_USEC* is a little more

difficult to describe. It is the time it takes the TLC5947 PWM counters to count from zero to 4095, thereby completing the 12-bit PWM cycle. The PWM oscillator internal to the TLC5947 runs at 4 MHz under normal operating conditions. The period of the 4 MHz oscillator times 4,096 equals 1,024 microseconds.

Since both of these processes must occur for every row of the LED matrix data, the total time between interrupts is their sum. The TimerOne library is used to cause a periodic interrupt at this frequency.

Finally, we define the frame buffer which contains the data used to drive the complete LED matrix:

```
// Frame buffer definition
#define NUMBER_OF_ROWS 8
#define BYTES_PER_ROW 36

// Frame buffer is 2D array of bytes
byte framebuffer[NUMBER_OF_ROWS][BYTES_PER_ROW];
```

Foreground code in a sketch would put data into the frame buffer that represents the pattern to be displayed, and the background code contained in the interrupt service routine (ISR) moves the data from the frame buffer to the hardware continuously. The LED matrix is updated

around 122 times per second by the ISR.

From the foreground's code perspective, it just sets pixels to specific colors and these colors magically appear on the LED matrix. It is not necessary to call any kind of show function to force a display update; it all happens automatically and in real time.

With this understanding, most of the code in the sketch should now be self-explanatory. As a demo, I coded a scrolling "Nuts and Volts" text message and a swirling rainbow pattern which alternate. If you build the demonstration hardware and run this sketch, you will see a bright vibrant display without any flicker whatsoever.

Going Further Yet

The demonstration hardware can control 64 RGB LEDs or 192 single color LEDs. If this is still insufficient, it should be possible to add up to eight additional rows of LEDs. It is also possible to chain TLC5947 chips. Sixteen rows of LEDs combined with two TLC chips would bring the total RGB LED count to 256 with a refresh rate of approximately 60 frames per second. Not too shabby for such a small amount of hardware. **NV**

JOIN TEAM SYNERGY MOON!

Synergy Space Explorers are the power that drives the Synergy Moon space program. We are Team Synergy Moon, building a mission to the moon that includes Micro Satellites, Lunar Rovers and a Lunar Lander. We're going into space this year, and we're going to the moon with our Google Lunar XPRIZE mission.



You will have the opportunity to participate in space research, exploration and development missions, which currently include our Google Lunar XPRIZE mission to the moon, the Artemis NanoSat Constellation project and our remote controlled Tesla Orbital Space Telescope.



Get on board now and be part of this great adventure!

DIY SATELLITES AND SPACECRAFT SYSTEMS
info@synergymoon.com

The RN4020 PICtail Plus BLE

Bluetooth has morphed yet again, and it seems that everybody wants Bluetooth on their iPhone or Android phone. These days, Bluetooth Low Energy (BLE or BTLE) is the new "must have" control and monitoring medium. The new crop of BLE radios are (as a whole) easy to use. Most every manufacturer's BLE radio entry has an associated firmware API (Application Interface). Many of the new BLE platforms employ simpler data interfaces which are based on good old RS-232. The iPhone and Android application creation barriers that have for too long impeded interactive BLE control are falling like hot rocks from a caveman's hands. In this edition of the Design Cycle, we will take a look at the latest BLE offering from Microchip.

RN4020

The Microchip RN4020 is a qualified and certified Bluetooth 4.1 radio module. The RN4020 takes its instructions via ASCII commands over a UART connection. Everything the RN4020 needs to transmit and receive is packed in under the module's shield. There are also under-the-hood provisions for analog and digital I/O. In that the RN4020 can operate alone under control of its internal scripting engine, a resource-rich microcontroller is

not required to assist the RN4020. A pair of connected RN4020s is perfectly capable of taking care of themselves with little or no help from outsiders.

Bringing Up Baby

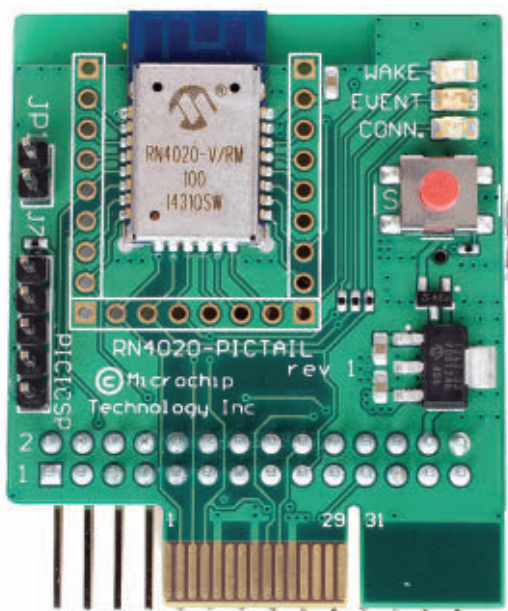
You are reading this column, which means you are also perfectly capable of reading the RN4020 datasheet. So, instead of spouting RN4020 technical specifications and I/O capabilities, let's discuss the RN4020 in the languages of ASCII and C. Our RN4020 hardware will be represented by the Microchip RN4020 PICtail which you can see under the lights in **Photo 1**.

BLE radios split up as centrals and peripherals. The peripheral radio advertises its connection status, while the central radio starts the connection process. When a pair of BLE radios connect, the next thing they do is bond. Once the radios have bonded, security items are saved and used for the next connection between the two devices. Bonded radios cannot cheat and connect to other devices.

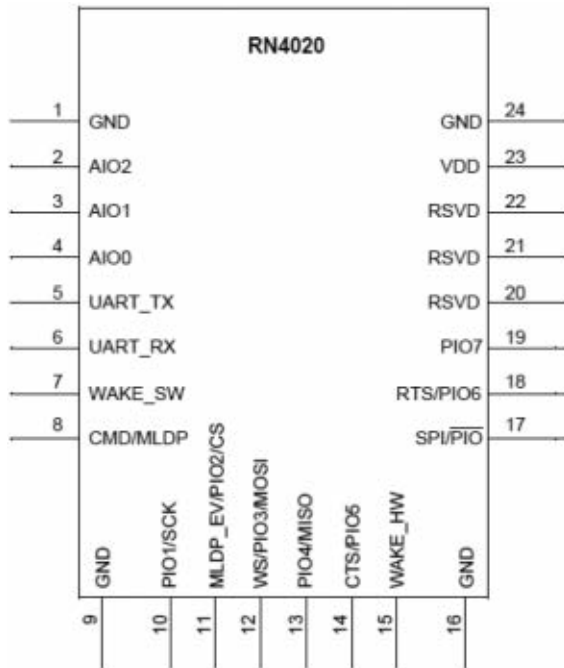
Controlling the RN4020

Figure 1 is a skeletal pinout of the RN4020 BLE radio module. The `WAKE_SW`, `CMD/MLDP`, and `WAKE_HW` pins are responsible for initiating RN4020 state changes. The status of the RN4020 is reported by three output pins (PIO1, PIO2, PIO3). Let's check out the various RN4020 states and their consequences beginning with the `WAKE_SW` pin.

The `WAKE_SW` pin (pin 7) controls the RN4020 operating state. When `WAKE_SW` is forced logically high, the RN4020 wakes up and enters Active mode. After being roused, the RN4020 will send "CMD" to the UART.



■ **Photo 1.** The RN4020 PICtail is designed to allow us to come up to speed quickly on the RN4020 hardware and API. Almost everything we need to evaluate the RN4020 is soldered onto the PICtail printed circuit board.

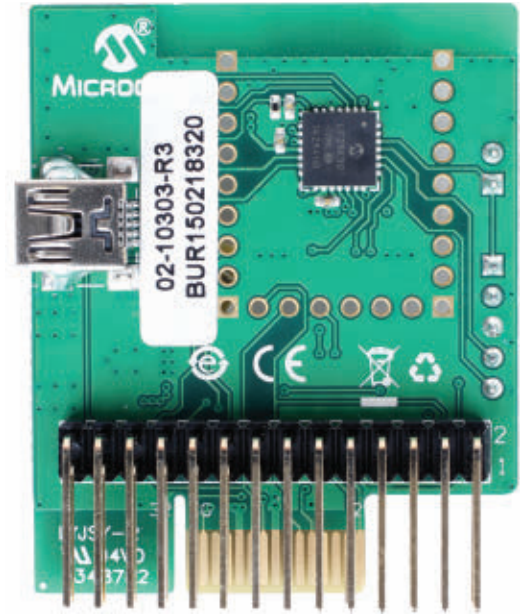


■ Figure 1. The RN4020 is small and so is the pin count. That's a good thing since the more pins we have to keep up with, the more code we have to write.

This signals that the RN4020 is in Command mode and ready to service commands coming in via the UART. When the WAKE_SW pin is returned to a logically low condition, Command mode is exited and “END” is sent to the UART. The RN4020 will then enter Deep Sleep mode. The MLDP_EV pin (pin 11) goes logically low to indicate Deep Sleep Mode. If the UART baud rate is set for 2400 bps, the UART is always accessible and the WAKE_SW pin does not need to be forced logically high to wake the RN4020.

The CMD/MLDP pin (pin 8) is used to control the RN4020 when the radio module is using the MLDP serial data service. Forcing the CMD/MLDP pin logically high puts the RN4020 into MLDP mode. In MLDP mode, all data from the UART is sent to the peer device as a datastream. This mode is useful for wire replacement applications. Setting the CMD/MLDP pin logically low will force an exit of MLDP mode, then the RN4020 returns to Command mode and sends “CMD” to the UART.

If the RN4020 is in Dormant mode, taking the WAKE_HW pin (pin 15) logically high will power up the RN4020. Following the power-up sequence, the RN4020 can be instructed to perform a factory reset. The factory reset is kicked off by flipping the WAKE_HW pin logically high, logically low, and logically high three times within five seconds. If the WAKE_SW pin is logically high during a factory reset operation, a complete factory reset is performed. If the WAKE_SW pin is logically low during the factory reset, a partial factory reset is performed. A partial factory reset preserves the device name, private service, and scripts. A private service is any data link configuration that is not a registered Bluetooth service. For instance, Microchip’s MLDP is a private service.



■ Photo 2. The PIC18LF25K50 is configured as a USB CDC device. This configuration allows the RN4020 PICtail to be attached directly to a PC’s USB port. Commands and data are passed between the RN4020 PICtail and PC using a simple terminal emulator program.

Pin PIO1 defaults to the CONNECTION LED pin and will present a logical high when the RN4020 is connected to a peer device. The MLDP_EV pin is used as an indicator in MLDP mode, and will go logically high when the RN4020 must output a status to the UART or requests a response from the host MCU. Active mode is indicated by driving the WS pin logically high.

The RN4020 PICtail Plus Board

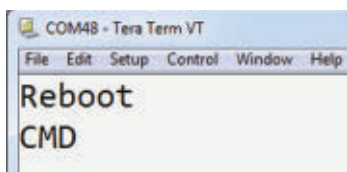
The business end of the RN4020 PICtail is exposed in **Photo 1**. As you can see, all of the RN4020’s I/O pins are accessible via a set of pads surrounding the radio module. The RN4020 PICtail also includes an onboard 3.3 volt voltage regulator, a user pushbutton, three status LEDs, an ICSP interface (J7), and an interface selection jumper (J1).

When jumper J1 is installed, the RN4020 PICtail’s edge connector interface is active. The edge connector mates with the Microchip Explorer 16 development board or any other dev board that supports the PICtail or PICtail Plus footprint. If the jumper at J1 is not installed, the

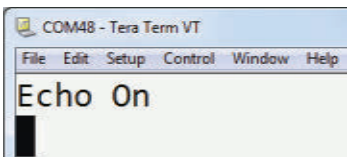
RESOURCES

CCS
CCS C Compiler
www.ccsinfo.com

Microchip
RN4020 PICtail
Explorer 16
www.microchip.com



■ Screenshot 1. Rebooting the RN4020 is one way to get the coveted "CMD" message.



■ Screenshot 2. By simply entering "+" we can see the commands as they are entered.

RN4020 via a simple terminal emulator like Tera Term Pro. In addition to the CDC code, the PIC18LF25K50 is loaded with microcode that exposes an optional PIC18 command shell that allows us to use a terminal emulator to manipulate the PIC's I/O pins.

What's Better than an RN4020 PICtail Plus Board?

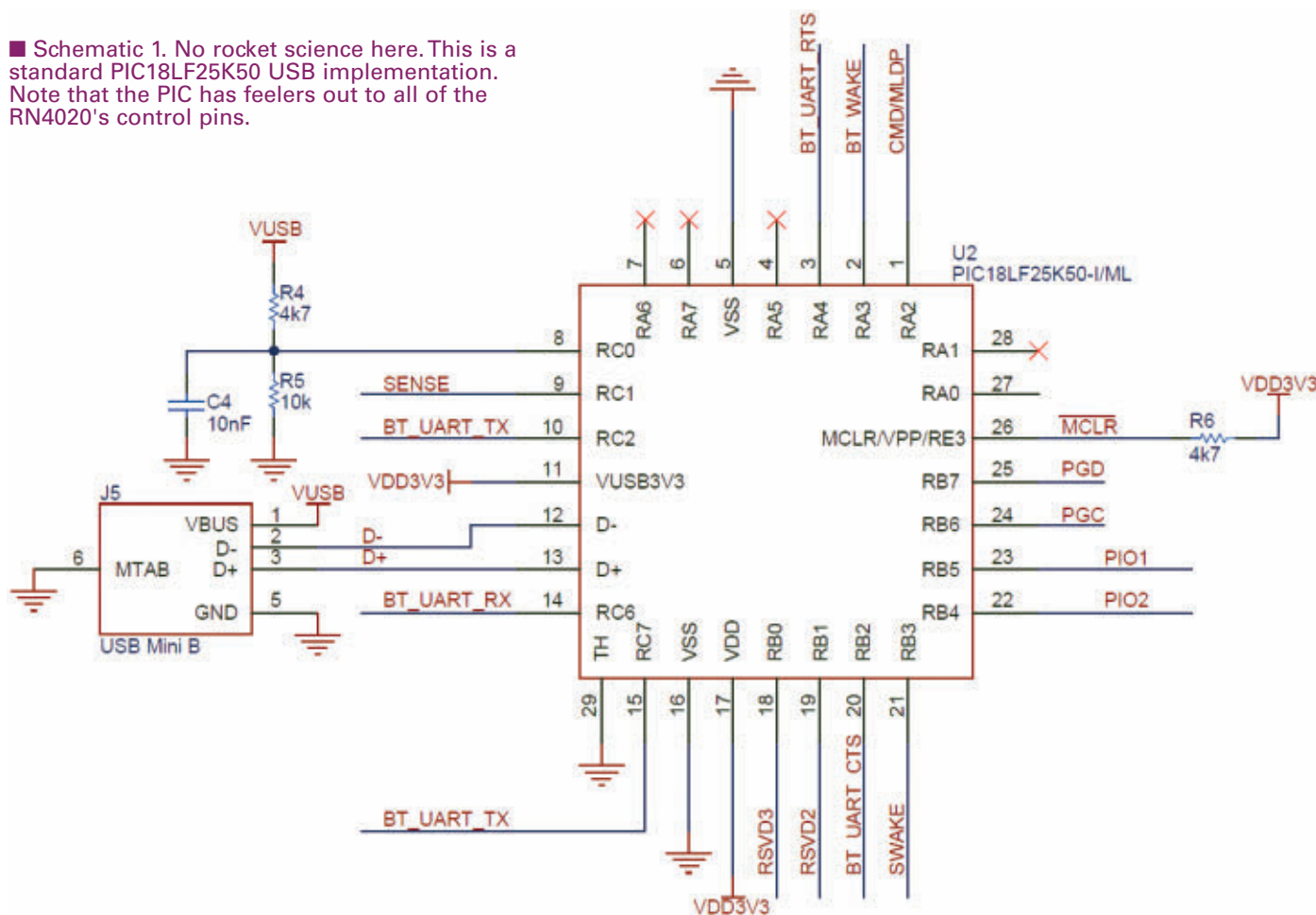
Two RN4020 PICtail Plus boards! Now that we have a basic knowledge of BLE and the RN4020, let's put those back-side USB portals to work and check out what it takes to build an RN4020 BLE link.

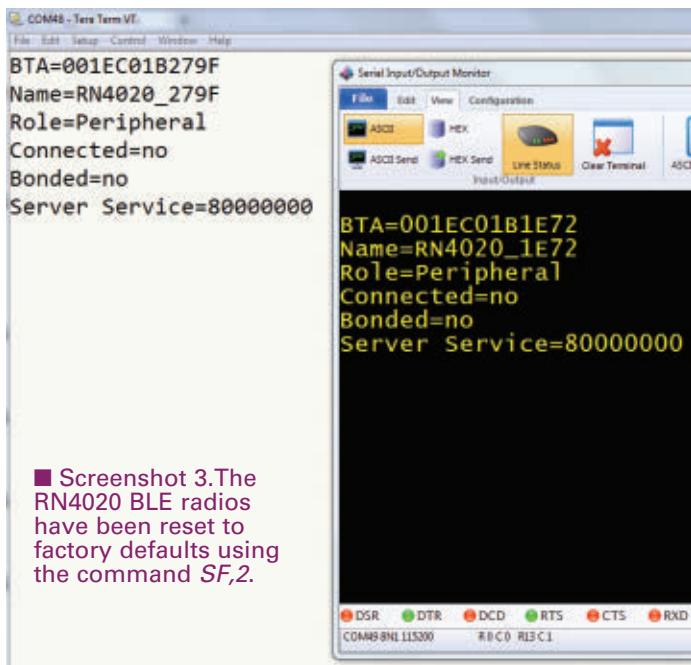
The RN4020 PICtail comes ready to rock at 115200 bps. A proper USB cable is also packed with your new RN4020 PICtail. Upon plugging in a brand new RN4020 PICtail and kicking off a Tera Term Pro terminal session, I was facing a blank terminal emulator window. So, just to see if the RN4020 was really there, I issued a Reboot command (*R,1 - Enter*). The results are shown in **Screenshot 1**. The next thing we should do here is make it a bit easier to see what we're doing. We can do this by simply entering "+" and hitting Enter. **Screenshot 2** assures

PICtail falls under the control of a USB-endowed PIC18LF25K50. The PIC18LF25K50 is shown in **Photo 2**, which is a shot of the other side of the RN4020 PICtail printed circuit board (PCB).

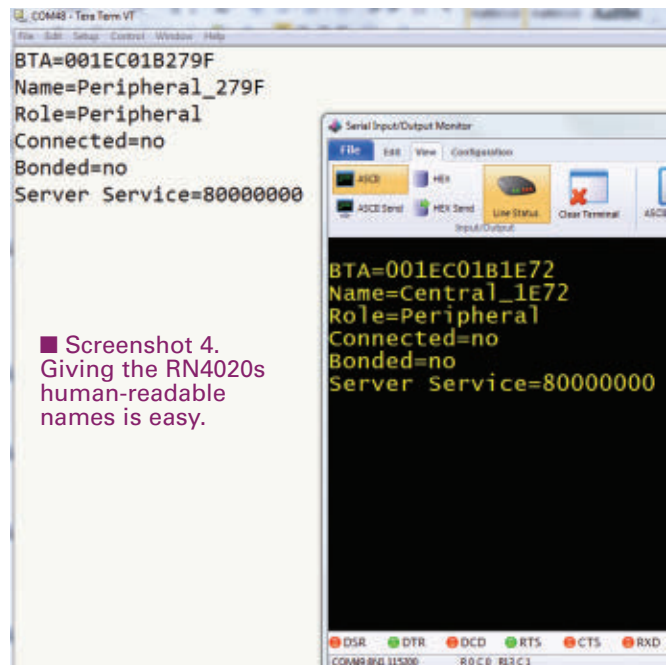
Schematic 1 tells us that the PIC18LF25K50 is acting as a USB CDC interface that logically links its USB portal with its UART component. The PIC's USB portal is intended to connect to a PC host USB port, while the PIC's UART is aimed at the RN4020's UART interface. This arrangement allows us to communicate with the

■ Schematic 1. No rocket science here. This is a standard PIC18LF25K50 USB implementation. Note that the PIC has feelers out to all of the RN4020's control pins.





■ Screenshot 3. The RN4020 BLE radios have been reset to factory defaults using the command *SF,2*.



■ Screenshot 4. Giving the RN4020s human-readable names is easy.

that our simple command was executed.

I issued the *D* command to obtain the contents of the Tera Term Pro and serial I/O monitor windows you see in **Screenshot 3**. Both of the RN4020s have been rebooted and reset to default factory settings. At this point, both RN4020 PICtails have the WAKE LED illuminated. As you've probably already figured out, we have a couple of RN4020 PICtails and there is a BLE link in our future.

Note that both of our RN4020 PICtails are unconnected/unbounded peripheral devices. To form a BLE link, we will need to configure one of these RN4020s as a central. Let's make the radio attached to the CCS C compiler serial I/O monitor the central; we'll call it "Central." To make that name stick, we must issue the command *S,Central*. Let's make the RN4020 attached to Tera Term Pro the peripheral and call it "Peripheral." Our work is checked in **Screenshot 4**.

Now, let's turn our attention to the central BLE radio which really isn't configured to be the central yet. Let's draw the king's sword and knight the RN4020 attached to the serial I/O monitor. In our case, the king's sword is the *SR* (Set Features) command. The argument of the *SR* command is a 32-bit bit mask. Here's the lowdown:

```
0x80000000 - Start the connection as a central
0x10000000 - Support MLDP
0x02000000 - Enable UART Flow Control
```

So, our command to create a central that supports MLDP with UART flow control is *SR,92000000*. A reboot (*R,1*) is required following the *SR* command. We can leave the Server Services (command *SS*) alone for now as its default argument of *0x80000000* is quite alright for us.

We now have a central. So, let's configure the RN4020 attached to Tera Term Pro as the peripheral. As

RF Specialists

"Making your RF ideas into profitable products."

FCC Part 90 Compliant

USX2 - NBFM Multi-Channel UHF Transceiver with Programmable RF Power

MURS (Multi-Use Radio Service)

SHX1 - Long Range, High Power MURS Band Transceiver

ZigBee Pro

OEM Modules and USB ZigBee Sticks, Mesh Networks

Industrial Bluetooth

OEM, Modules, Wireless Device Servers, RS-232 Long range options, low cost

Ultra-Low Power Wi-Fi networking module and Eval board

The AMW006 'Numbat' module is an ultra-low power Wi-Fi networking module with full regulatory certification.

RF Design Services

Prepared to work with your in-house engineers, or support your RF project from initial design to implementation.

Industrial • Military • Space • Medical • Smart Grid Metering • SCADA • Lighting Control

LEMOS INTERNATIONAL

Tel: 1.866.345.3667
orders@lemosint.com
www.lemosint.com

you would imagine, the command sequence to produce a peripheral pawn is very similar to that of the central. The bit mask forms up like this:

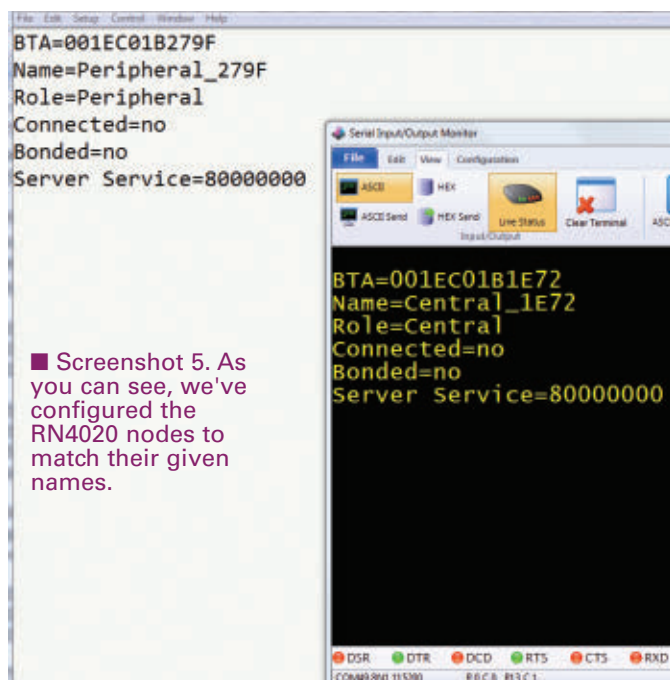
```
0x20000000 - Auto Advertise
0x10000000 - Support MLDP
0x02000000 - Enable UART Flow Control
```

The command *SR,32000000* plus a reboot will enable our peripheral RN4020. Since only politicians say “trust me,” I’d rather produce the real proof like what you see in **Screenshot 5**.

We are ready to attempt a connection. The central is set up to scan for advertisements from the peripheral on command. The peripheral knows to start advertising on power-up. With that, we should be able to issue the *F* (Start Scanning) command at the central and locate the advertising peripheral RN4020 node. According to **Screenshot 6**, we found an RN4020 peripheral node. A peek at the BTA (Bluetooth Address) tells us that this is our peripheral node, which again is called Peripheral. The BTA is also referred to as the MAC address. To stop the scanning, we issue the *X* (Stop Scan) command which is obscured by an AOK response in **Screenshot 6**.

Once the scanning has ceased, we can proceed to connect to Peripheral by issuing the *E* (Establish Connection) command. The *E* command syntax is also overwritten in **Screenshot 7**. So, here’s what was entered in the serial I/O monitor terminal emulator window:

```
E,0,001EC01B279F
```



■ Screenshot 5. As you can see, we've configured the RN4020 nodes to match their given names.

The zero following the command identifies the following MAC address as public instead of random. The WAKE and CONN LEDs on both the central and peripheral RN4020 nodes are illuminated. Now what?

Since we don't have an application in place, this is a good time to test drive MLDP. We can invoke MLDP mode by simply entering the command *I* at the central. Once MLDP mode is active, the peripheral will acknowledge the mode change and follow the lead of the central. Messages can then be sent between the nodes in streams as shown in **Screenshot 8**. The central message welcomes the peripheral to MLDP. The peripheral RN4020 must have some kin folk in Tennessee. The peripheral's answer is straight out of the Grand Ole Opry and Minnie Pearl's mouth.

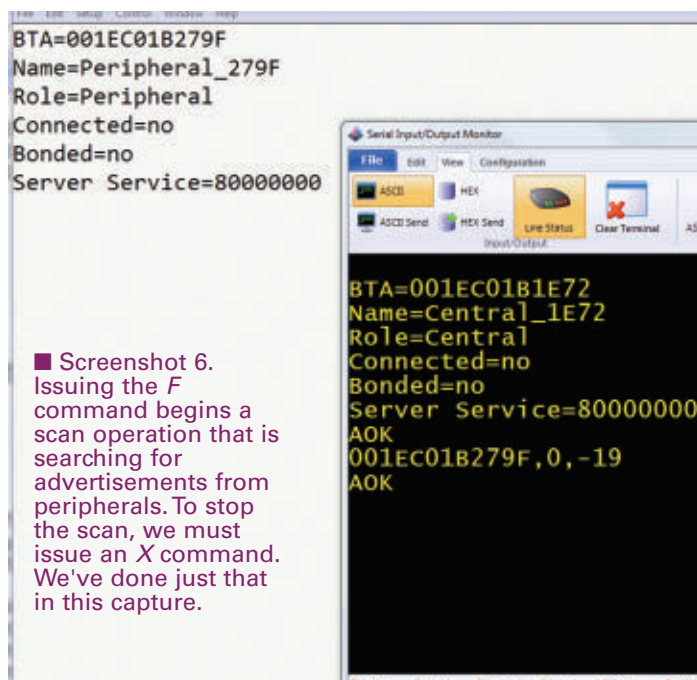
RN4020 How-To

To continue on with the RN4020, you will need to have some BLE 4.0 devices available to you. Microchip has provided demo code for both BLE-equipped Android devices and their associated PIC-based peripherals. However, you already have enough to get started on your own.

For instance, you can use the CCS C compiler to help you. Let's “wake up” the RN4020:

```
output_high(PIN_WAKESW);
```

Now that the RN4020 is awake, let's perform a factory reset and give the RN4020 some time to get its act together:



■ Screenshot 6. Issuing the *F* command begins a scan operation that is searching for advertisements from peripherals. To stop the scan, we must issue an *X* command. We've done just that in this capture.

```
printf("SF,2\r\n");
delay_ms(500);
```

We'll make this RN4020 node the central. Before we make that official, let's give it an appropriate name:

```
printf("S-,IamCentral\r\n");
```

Okay. Now, let's sign the papers that declare this RN4020 node a central, give it MLDP powers, and UART flow control support:

```
printf("SR,92000000\r\n");
```

Don't forget to reboot:

```
printf("R,1\r\n");
```

I think you get the idea. Using the RN4020 PICtail, you can "emulate" all of the actual embedded commands that you would issue using a PIC. You can retrieve messages from the RN4020 using either a polling method or interrupt routine. If you only need to capture single character sequences, the CCS C compiler's *kbhit* function can be used in your RS-232 receive routines.

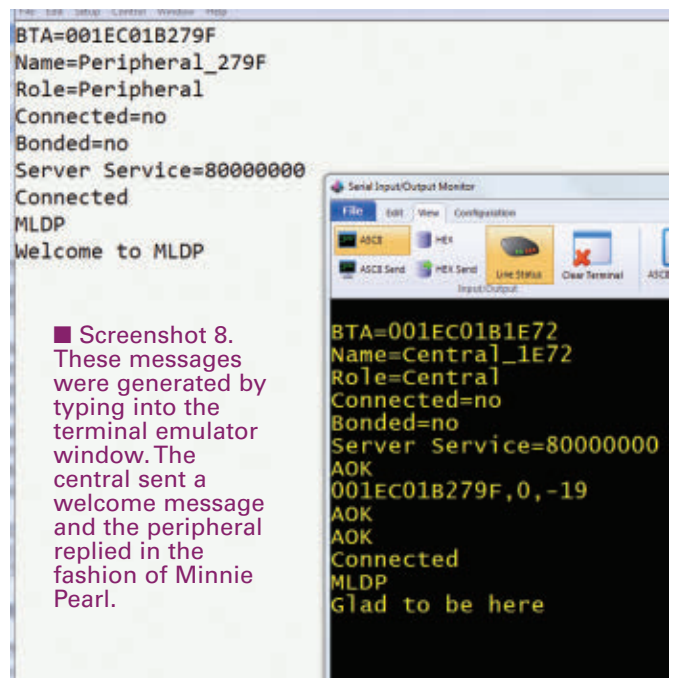
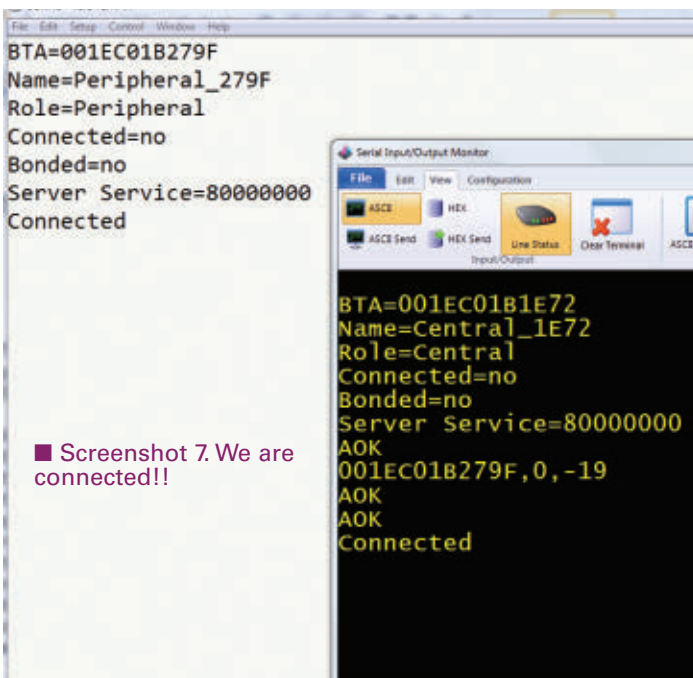
For those applications where you can't afford to miss a character or the characters are coming in while you're out plowing the North 40, this code might be useful:

```
/**
 * USART DEFINITIONS
 */
```

```
#define USART_RX_BUFFER_SIZE 256
#define USART_RX_BUFFER_MASK
( USART_RX_BUFFER_SIZE - 1 )
unsigned int8 USART_RxBuf[USART_RX_BUFFER_SIZE];
unsigned int8 USART_RxHead,USART_RxTail;

#INT_RDA
void RDA_isr(void)
{
    unsigned char data;
    unsigned char tmphead;

    //read the received data
    data = getc();
    //calculate buffer index
    tmphead = ( USART_RxHead + 1 ) &
USART_RX_BUFFER_MASK;
    //store new index
    USART_RxHead = tmphead;
    //handle buffer overrun
    if ( tmphead == USART_RxTail )
    {
        CREN = 0;
        CREN = 1;
        USART_RxTail = 0x00;
        USART_RxHead = 0x00;
    }
    // store received data in buffer
    USART_RxBuf[tmphead] = data;
}
```




```

//*****
/**  RETRIEVE A CHARACTER FROM USART
//*****
unsigned int8 recvchar(void)
{
    unsigned char tmptail;
    /* wait for incoming data */
    while ( USART_RxHead == USART_RxTail );
    /* calculate buffer index */
    tmptail = ( USART_RxTail + 1 ) &
USART_RX_BUFFER_MASK;
    USART_RxTail = tmptail;
    /* store new index */

    return USART_RxBuf[tmptail];
    /* return data */
}
//*****
/**  CHECK FOR CHARACTER IN RING BUFFER
//*****
unsigned int8 CharInQueue(void)
{
    return(USART_RxHead != USART_RxTail);
}

```

The interrupt code I've listed can buffer up to 256 incoming characters at 115200 bps. To check for a character in the buffer, call the *CharInQueue* function. If the *CharInQueue* function returns a TRUE, use the *recvchar* function to remove a byte from the buffer. Assign a UART to the interrupt routine using the *#use RS232* preprocessor:

```
#use rs232(baud=115200,parity=N,
xmit=PIN_C6,rcv=PIN_C7,bits=8)
```

The RN4020 PICtail is a great way to get started with your own scratch design. In addition to the edge connector, you can get at the RN4020 using the PICtail's male header pins.

This makes the RN4020 available to you via a perfboard, which makes it super easy to add the RN4020 to your Design Cycle. **NV**

The Convenient All-in-One Solution for Custom-Designed Front Panels & Enclosures



**FREE
Software**



ONLY \$90.24
with custom
logo engraving

You design it
to your specifications using
our FREE CAD software,
Front Panel Designer

We machine it
and ship to you a
professionally finished product,
no minimum quantity required

- Cost effective prototypes and production runs with no setup charges
- Powder-coated and anodized finishes in various colors
- Select from aluminum, acrylic or provide your own material
- Standard lead time in 5 days or express manufacturing in 3 or 1 days

**FRONT PANEL
EXPRESS**

FrontPanelExpress.com
1(800)FPE-9060



AP CIRCUITS
PCB Fabrication Since 1984

As low as...

\$9.95
each!

Two Boards
Two Layers
Two Masks
One Legend

Unmasked boards ship next day!

www.apcircuits.com



CELEBRATING *a DECADE of* MAKING

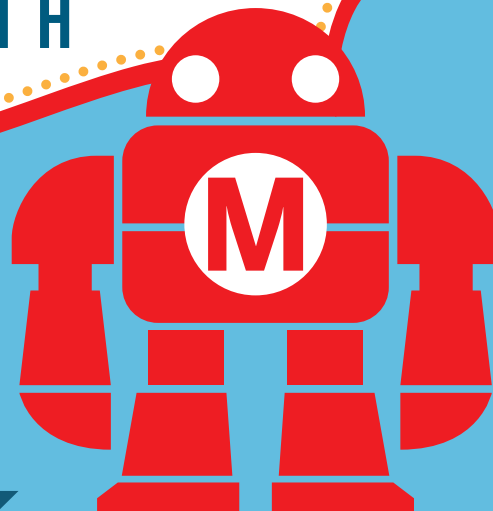
10th ANNUAL BAY AREA

GREATEST SHOW & TELL ON EARTH

MAY 16+17

SATURDAY 10AM-8PM
SUNDAY 10AM-6PM

makerfaire.com



Brought to you by
Make: magazine

Maker Faire®

PRESENTED BY 

GOLDSMITH
SPONSORS

 AUTODESK

 iROBOT®
DRIVE THE FUTURE

 ITCA®
ITALIAN TRADE AGENCY

SILVERMITH
SPONSORS

 NVIDIA

 ROBLOX

ELECTRONET

CORIDIUM
Floating point **BASIC** for
ARM controllers from \$5.00
www.coridium.us

ALL ELECTRONICS
CORPORATION
Electronic Parts
and Supplies.
www.allelectronics.com
Free 96 page catalog 1-800-826-5432

For the ElectroNet
online, go to
www.nutsvolts.com
click **Electro-Net**

USB Add USB to your next project--
It's easier than you might think!
USB-FIFO • USB-UART • USB/Microcontroller Boards
RFID Readers • Design/Manufacturing Services Available
Absolutely NO driver software development required!
www.dlpdesign.com

SERVOCITY™ \$299.99
6-14V HEAVY DUTY
Linear Servos
Three speeds to choose from!
to controller
to battery
Plug & Play

HOBBY ENGINEERING
Kits, Parts and Supplies
www.HobbyEngineering.com

PCB, PCBA
and More!
Myro
Low cost
High Quality
www.myropcb.com
1-888-PCB-MYRO

Ironwood ELECTRONICS
75 GHz Sockets
High Speed Microwave Applications
RoHS
1-800-404-0204
www.ironwoodelectronics.com

superbrightleds.com
COMPONENT LEDs • LED BULBS • LED ACCENT LIGHTS

\$3
2 Layers ea.
• FULL DRC CHECK
• CAD/CAM REVIEW
• SOLDER MASK ON BOTH SIDES
Accurate inc.
www.PCB4u.com
(408) 748-9600

ramsey
www.ramseykits.com
AM/FM Broadcasters • Hobby Kits
Learning Kits • Test Equipment
...AND LOTS OF NEAT STUFF!
GET THE NUTS&VOLTS DISCOUNT!
Mention or enter coupon code **NVRMZ142**
and receive 10% off your order!

INVEST in your BOT!
Hitec
12115 Paine Street • Poway, CA 92064 • 858-748-6948 • www.hitecrod.com

MOBILE APP NOW AVAILABLE!
SERVO
Download NOW On Your
Favorite Mobile Device!
FOR ROBOT BUILDERS
iOS • ANDROID • KINDLE FIRE

Wanna learn more about electronics?
www.nutsvolts.com

3D Printed Custom Storage Boxes

*3D printers are the wave of the future — or so countless articles, reviews, breathless news commentators, and, of course, the machine's manufacturers keep telling us. By now, we've all seen a plethora of itty-bitty cubes, Yoda heads, chess pieces, interlocking gears, and other interesting but ultimately useless "things" created with 3D printers. Though many of these little demo pieces are impressive by themselves, they never quite cross over into the realm of *useful.* As we believe the "what is 3D printing" topic has been done to death, we thought it was high time to bring you a useful series on how to actually implement 3D printing. Specifically, working with 3D printers and showing you how to use them for practical projects on your workbench.*

I've found many uses for my da Vinci 3D printer, but recently saw a design on Thingiverse.com that piqued my interest. It was a small storage box with drawers (refer to Figure 1). Since then, I've seen many different storage boxes for electronics including pill boxes and large plastic cases with drawers, but one thing was always wrong with them. Either they didn't fit all the parts I had or they required lots of inserts to make the drawers fit the parts. Why couldn't someone just design a box especially for electronic components? With my 3D printer, I realized I could.



■ FIGURE 1.
Resistor box.

I started with a design from Thingiverse.com and imported it into Tinkercad design software (which is free at Tinkercad.com). I then went to work making drawers that fit what I wanted. My thought was to make a box to contain all the parts for one design. Then, when I want to build that design, I could just pull out the box and go to work.

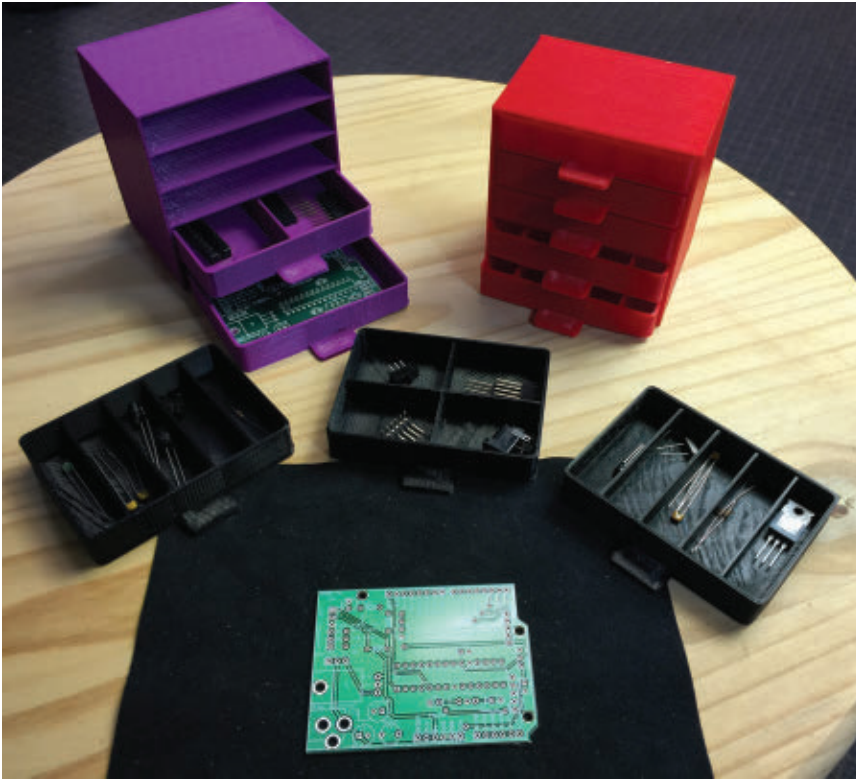
The first box I created held leaded devices like resistors and capacitors, but it could just have easily held strips of surface-mount components. The difference was it took up way less space and I could print any arrangement I wanted. Because the boxes were so small, I could glue them together for even more customization. I had one larger parts drawer that I purchased at Home Depot and stored all the parts to build a few Microchip PIC based CHIPINO modules, but the size was far larger than I needed. So, I decided to create custom drawers that could hold everything from the circuit board all the way down to the individual LEDs and 1/8 watt resistors. When I was

done, it saved an incredible amount of space.

The purple box shown in **Figure 2** contained four different drawer layouts. I was able to fit enough



■ FIGURE 2. Large drawer storage vs. small drawer storage.



■ **FIGURE 3. CHIPINO module storage box.**

components in the drawers for at least five CHIPINO modules.

Figure 3 shows some of the drawers. One held just the circuit board, while the larger 28-pin PIC16F886 and 28-pin DIP sockets were in a separate drawer that was

divided into two sections. I then made a four-section drawer that held some of the larger components. The final two drawers had five sections each to hold resistors, diodes, regulators, capacitors, and many other parts. In the end, all the parts were contained in one small box with five drawers.

The other advantage to this setup was I could customize the box for more drawers or wider drawers as needed because I owned the files and could print anything I needed on my printer.

I could also use a little super glue (or better yet, acetone) to fuse the boxes together to make my own custom size set of drawers. If I need one large drawer, I could do that too. If I need a handy way to carry the boxes instead of gluing them together, I can get a wood case like the one in **Figure 4** that I picked up from Michael's craft store for \$4. It has handles on the side for carrying, and large openings for drawers. I 3D printed some large drawers that fit the bottom row and then put the small box of drawers in the upper slots. This way, I could keep designs stored away in an organized

manner, rather than just components.

This is only limited by your imagination because you can customize the storage boxes any way you want once you have a 3D printer.

Another option is to 3D print the part number in the bottom of the box. That way, when the slot is empty, you know exactly what was in there. You could just put a paper note or sticker in the bottom instead, but 3D printing the part number is free and it will never get lost.

Now, add to this all the screws, nuts, and stand-offs you may have in your collection of electronic components. If we used those large box drawers for all the various components, we'd have more boxes than space. Because of that, I have all the hardware collected in one drawer. Then, I have to sort through it to find the part I need.

Being able to quickly build a custom drawer size that is much smaller allows me to organize the various hardware and take up less space.

ARLCD 3.5-inch Arduino GPU Combo Just \$99.00



Product Specifications: 3.5" color TFT LCD • 320 x 240 Resolution • 65k colors • Touchscreen • Powerful 16-bit microcontroller GPU • 4MB flash memory for storing fonts, bitmaps & macros • USB 2.0 • Overall outline dimensions: 3 x 3 x .9 inches • 6-9V operating voltage • Extremely low power - draws less than 200mA • ARLCD Arduino GUI Library • Arduino UNO R3 Compatible



I can also use different color plastics to make the drawers and boxes. Red can be resistors, blue for capacitors, black for hardware, etc. The options are endless.

I've gotten to the point where I spend half my time sorting through all the various storage containers and boxes that hold my large collection of electronic parts rather than building electronic projects. So, over time, I hope to 3D print a whole new storage system for my lab. It should take up a whole lot less space.

I put my original drawer design up on my Thingiverse account so anybody can download the files and print them. I also made the designs public on Tinkercad so anybody can modify them to fit their specific needs.

I have a YouTube video showing other 3D print projects. You can see them at www.youtube.com/user/beginnerelectronics.

If you have a 3D printer question or project idea, send me an email at chuck@elproducts.com and I'll try to help. **NV**

Resources

Check out my website and blog:
www.elproducts.com

Check out my
YouTube Channel:
www.youtube.com/user/beginnerelectronics

Check out my 3D designs:
www.thingiverse.com/elproducts/designs

Tinkercad:
www.tinkercad.com

da Vinci 3D:
us.xyzPrinting.com

da Vinci Forums:
www.solidforum.com
www.voltivo.com



■ **FIGURE 4.** Wood storage box from Michael's craft store.



Introducing the ALL NEW TRONIX 1 Lab. Bold and exciting full color illustrations that include interactive content specific to each lesson.

This exciting new curriculum is the perfect tool for your classroom, and with the full online interactive activities that bring each project alive right before your eyes.

SHIPPING NOW!!!

*TRONIX 1 is backwards compatible with Tron.ix 1 and Mr. Circuit curriculums

CHECK OUT OUR FULL COLOR KITS


Our entire line of Solder Kits have been completely revised and updated to include beautiful full color illustrations.

Order your **FREE SAMPLE KIT TODAY**

LFComponents.com/free

615-625-2885

LFComponents.com



CubeSats — Part 3: Attitude and Velocity

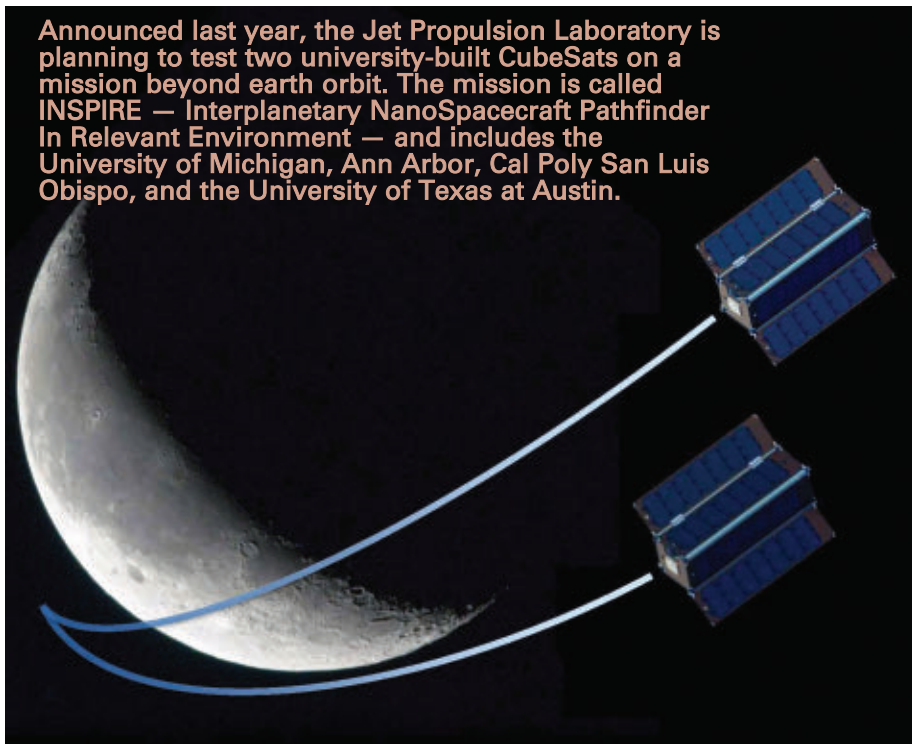
What's available for CubeSats is just amazing. I looked into them several years ago and didn't see nearly as much as I do today. From reader's comments, I've learned that there are even CubeSat materials being developed every month. CubeSats are going to open the solar system to DIY space exploration within our lifetime. So, this month, let's learn about some toys that permit CubeSats to control their attitude and velocity.

Spacecraft Attitude

Attitude refers to the orientation of a spacecraft in space. The first spacecraft, Sputnik 1 didn't maintain an attitude at all; it just tumbled as it orbited earth. Some spacecraft, on the other hand, were intentionally spun during launch like America's Explorer 1. The spin imparted an angular momentum that prevented the spacecraft from tumbling.

Some spacecraft maintained their attitude in three axes using systems like thrusters (this is called three-axis stabilization). Examples of three-axis stabilized spacecraft include the Voyager 1 and Voyager 2.

Announced last year, the Jet Propulsion Laboratory is planning to test two university-built CubeSats on a mission beyond earth orbit. The mission is called INSPIRE — Interplanetary NanoSpacecraft Pathfinder In Relevant Environment — and includes the University of Michigan, Ann Arbor, Cal Poly San Luis Obispo, and the University of Texas at Austin.



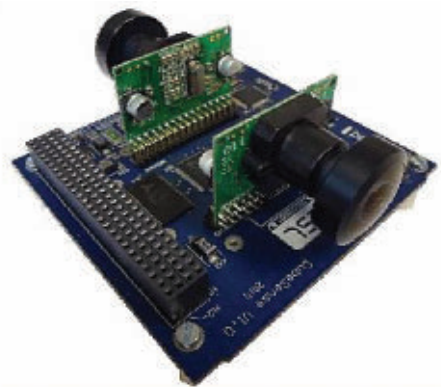
Methods to maintain stabilization are only half the story, however. A spacecraft must first determine its orientation in space. Although this article has broken these systems into two parts — attitude control and attitude determination — satellites combine the two operations into a single subsystem called the Attitude Determination and Control System (ADCS).

CubeSat Attitude Determination

CubeSats need an outside reference in order to determine their attitude. Outside references include things like the earth's horizon, the sun, stars, and earth's magnetic field. Sun sensors are conceptually simple; they're circuits that determine how

closely a surface faces into the sun.

The shortcoming is that a sun sensor can only provide a position along a single axis in space.



This is the CubeSense module by the University of Stellenbosch, South Africa that combines a sun and earth sensor. One CMOS imager detects the sun, while the other detects the earth's horizon.

Post comments on this article and find any associated files and/or downloads at www.nutsvolts.com/index.php?/magazine/article/april2015_NearSpace.

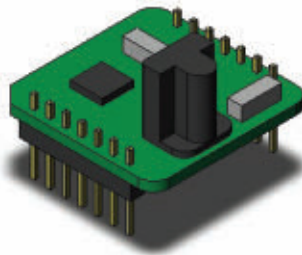
Therefore, a CubeSat carrying only a sun sensor could be rotated at any angle along the axis connecting the CubeSat to the sun and the CubeSat would not know. To increase attitude knowledge, a second or even third attitude sensor is sometimes incorporated into the CubeSat.

By using an imager, a sun sensor permits software to determine its offset from the sun in terms of a vector (magnitude and angle). To ensure the sun sensor only detects the sun, there's a neutral density filter over the sun sensor's imager that's so dense that only sunlight can reach it.

CubeSats in earth orbit can use earth itself as the second reference point. The earth sensor measures the position of earth's horizon with respect to the CubeSat. That's made possible because there's a huge difference in temperature between the warm earth and cold space. By measuring the position of earth's horizon (in infrared), a CubeSat knows the location of the horizon with respect to the CubeSat and therefore the nadir (straight down to the earth).

Earth-orbiting spacecraft (including CubeSats) can rely on magnetometers as another tool for determining their attitude. This means magnetometers are less effective the farther from earth the CubeSat orbits, and they're useless for interplanetary missions. If a magnetometer is being used for attitude determination, it's important that there be no distorting magnetic fields associated with the CubeSat. The fields can confuse the magnetometer and result in a bad attitude determination.

Two ways to get around this issue are to either place the magnetometer on the end of a boom that's a good distance away from the magnetically dirty spacecraft, or to determine the magnetic environment of the CubeSat and then subtract this as background noise from the magnetometer's signal. The second



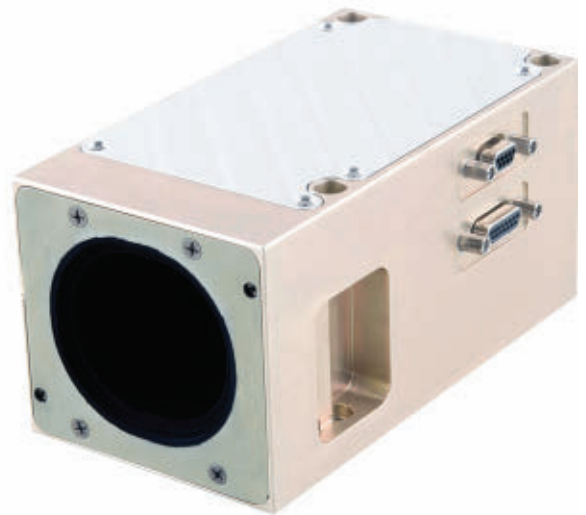
Magnetometers can be pretty simple devices because of today's electronic technology. This one is an engineering model of the MiniMag 3.

option is made more difficult by the changing magnetic environment of the CubeSat that results from systems and sensors being shut on and off.

One of the more interesting techniques for attitude determination is the star sensor. Simple ones were used as far back as 1964 for the Mariner 4 mission to Mars. Today, star sensors are not just for NASA and ESA spacecraft. Blue Canyon Technologies makes a Nano Star Tracker that determines the attitude of a CubeSat in three axes by comparing the images of star fields that it sees to a catalog of stored stellar positions. By identifying fields of stars, the software can determine the pointing direction and rotation of the CubeSat. Now, that's pretty cool.

CubeSat Attitude Control

Once a CubeSat's attitude is known, it can apply forces to change it to the desired attitude. Two popular ways CubeSats control their attitude are through magnetorquers and reaction wheels. Thrusters are also possible, but that requires propellant which can be in limited supply or non-existent in CubeSats.



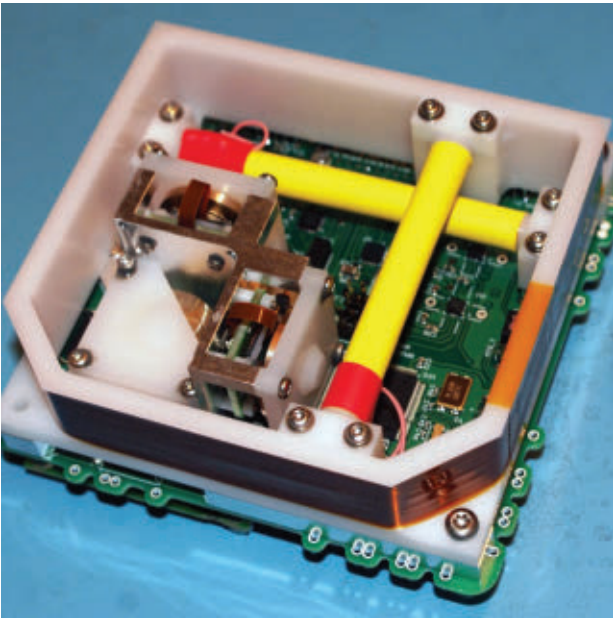
Weighing only 3/4 lb, the BCT Nano Star Tracker only occupies a quarter of the volume of a 1U CubeSat while only consuming 3/4W of power. A CubeSat carrying one of these will know its attitude in space to less than one degree.

Magnetorquers

Because of earth's native magnetic field, CubeSats in earth orbit cannot only determine their attitude with reference to it, they can also use it to change their attitude in space.

CubeSats can use magnetorquers or coils of wire wrapped around metal rods to push and pull on earth's magnetic field. Applying a current to a magnetorquer generates a magnetic field that interacts with earth's magnetic field. When the two fields don't line up, the misalignment creates a torque on earth and the CubeSat according to Newton's Third Law of Motion. Since the CubeSat is far less massive than earth, the CubeSat's orientation changes significantly while earth's doesn't. In other words, a CubeSat behaves like a compass needle.

CubeSats can carry two or three perpendicular magnetorquers to hold their attitude at precise angles. A magnetorquer is a great method to change the orientation of a CubeSat since it allows it to change its attitude



An example of two perpendicular magnetorquer rods. This particular example is from Delfi Space and will be used by the CubeSat to dump excess momentum. Delfi Space is the CubeSat program of Delft University of Technology (yeah, go Dutch!).

frequently without consuming propellant. Its weakness is that they still permit a CubeSat to spin along a magnetic field line.

Reaction Wheels

Reaction wheels are even better at holding a CubeSat attitude. They are motor mounted masses that are spun forwards or backwards. According to Newton's Laws, when an internal mass rotates, the CubeSat to which the motor and weight are attached spins in the opposite direction. That also means if a CubeSat is already spinning, a reaction wheel can slow that spin down by imparting a spin in the opposite direction on the CubeSat.

To control the spin and orientation of a CubeSat along three axes, three reaction wheels are installed into the CubeSat — each with its spin axis perpendicular to the other two. A fourth reaction wheel with a spin axis equidistant from the other three reaction wheels can give



A reaction wheel for CubeSats. As the wheel spins, the CubeSat spins in the opposite direction (according to Mr. Newton).

a level of redundancy in case one wheel fails.

There comes a time when the reaction wheels may be spinning at their highest designed speed. When

that happens, the wheels become saturated with momentum and must be desaturated to be of any further use. A method used to desaturate reaction wheels is to slow them down by transferring momentum to the CubeSat. Then, the CubeSat applies the magnetorquers in order to reduce the CubeSat's spin.

Changing CubeSat Velocity

I came across two interesting ways that CubeSats change their velocity: through the use of propulsion systems and braking systems

Propulsion

Among the many important factors of propulsion systems, two factors stand out: a given engine's efficiency and the total amount of velocity change possible. Specific Impulse (Isp) is one measure of the fuel efficiency of a propulsion system.

The unit of Isp is the second and it calculates the force generated by a propulsion system by multiplying the engine's Isp by the rate of propellant flow.

For example, an engine with an Isp of 100 seconds that consumes two pounds of propellant per second generates $100 \text{ seconds} \times 2 \text{ pounds/second}$, or 200 pounds of thrust. For comparison, black powder model rocket engines have an Isp between 70 and 100 seconds, while the Space Shuttle Main Engines (SSME) have an Isp of over

400 seconds in a vacuum.

Changes in velocity are often referred to as delta-V (ΔV). The maximum amount of ΔV possible is one factor in a spacecraft's useful life, and it depends on both the engine's efficiency and the amount of propellant carried by the spacecraft. What one notices about Isp is that the lower an engine's Isp, the more propellant a satellite must carry to achieve its desired maximum ΔV .

Because of a CubeSat's small volume and mass, its propulsion system must have a high Isp if it is to be able to affect the CubeSat's orbit in a serious way. The most efficient engines are those with the highest exhaust velocities which can be related to the temperature of the exhaust.

With specific impulses measured in the thousands of seconds, ion are the best engines we have for interplanetary travel. Since ion engines are stingy when it comes to propellant flow, they're unable to generate large amounts of thrust and are therefore incapable of launching rockets from the surface of a planet.

I found an ion engine for CubeSats called a pulsed plasma thruster (PPT) that's sold by Clyde Space, and has an Isp of 608 seconds. According to their website,

this engine can double the orbital lifetime of a CubeSat in some cases (the extension depends on the height of the CubeSat's orbit).

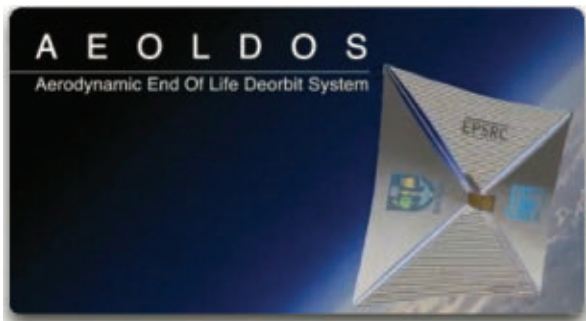
In high altitude orbits where the air density is low, a PPT only requires 0.04 watts of power to counteract the effects of air drag. In a lower earth orbit, the increased air drag means a CubeSat carrying a PPT needs to expend over two watts of power to overcome the effects of drag.

Another CubeSat propulsion system is the resistojet. Surrey Satellite Technology Ltd (SSTL) sells a resistojet consisting of a hot wire in which a liquid propellant is forced into contact with it. As the liquid gets hot, it vaporizes and creates thrust.

What kinds of propellant are used? The propellant used in the Surry Training, Research, and Nanosatellite Demonstrator (STRaND-1) CubeSat uses butane as its propellant. The system is called BPS for Butane Propulsion System and has an Isp of 90 seconds. With the amount of propellant available and the design of the resistojet, STRaND-1 can change its velocity by a total of two meters per second.

Braking Systems

I found another interesting product for CubeSats: braking sails. A concern for the astronautics community is the risk to expensive



An artist impression of the Aerodynamic End Of Life Deorbit System for CubeSats doing its thing. The sails are folded up inside a thin module attached to a CubeSat. At the appropriate time, the sails deploy out from the CubeSat.



This artwork by the University of Michigan depicts a test of the ion engine they are designing. This engine uses permanent magnets in order to reduce the power requirements of the engine.

satellites that a boom of CubeSat launches could create. Orbits crowded with large numbers of inexpensive CubeSats create collision risks and no one with a \$20 million satellite wants their hardware damaged by a \$20,000 CubeSat. One way to mitigate this risk is to place CubeSats into low earth orbits (LEO) that decay after a year or two. A second way is to install braking systems on CubeSats.

The orbital lifetime of a spacecraft depends on its size, shape, mass, and altitude of its orbit. The lowest orbiting CubeSats remain in orbit for less than a week, while higher altitude ones can remain in orbit for over five years.

Once a CubeSat has completed its mission, there's usually no reason for it to remain in orbit. Therefore, an end of life system to return CubeSats back to the atmosphere in a destructive manner fills an important need.

Clyde Space is

creating AEOLDOS, or Aerodynamic End Of Life Deorbit System for CubeSats as a way to remove CubeSats from orbit. AEOLDOS is a system to de-orbit CubeSats, as Clyde Space puts it. It acts like the reverse of a solar sail.

Instead of using solar radiation pressure to propel a CubeSat, the braking sail increases the surface area of a CubeSat in order to slow it down. The increased surface area increases the drag that the atmosphere imparts to the CubeSat. The lower the CubeSat's altitude, the greater the density of earth's atmosphere and the greater the drag created by the sail. The braking sail will remove a CubeSat from orbit much more quickly than it would otherwise.

This was a brief discussion about attitude control and change of velocity systems available for CubeSats. Next time, I'd like to acquaint readers with some of the CubeSat programs out there. You'd be surprised to learn where some people want to send CubeSats.

Onwards and Upwards,
Your near space guide **NV**

The Nuts & Volts **WEBSTORE**

GREAT FOR DIYers!

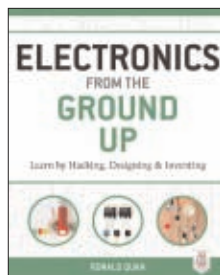
Electronics from the Ground Up: Learn by Hacking, Designing, and Inventing

by
Ronald Quan

Are you fascinated by the power of even the smallest electronic device? Electronics from the Ground Up guides you through step-by-step experiments that reveal how electronic circuits function so you can advance your skills and design custom circuits. You'll work with a range of circuits and signals related to optical emitters and receivers, audio, oscillators, and video.

Paper back 544 pages.

\$30.00



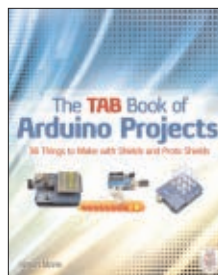
The TAB Book of Arduino Projects

by Simon Monk

The ultimate collection of DIY Arduino projects!

In this easy-to-follow book, electronics guru Simon Monk shows you how to create a wide variety of fun and functional gadgets with the Arduino Uno and Leonardo boards. Filled with step-by-step instructions and detailed illustrations, The TAB Book of Arduino Projects: 36 Things to Make with Shields and Proto Shields provides a cost estimate, difficulty level, and list of required components for each project.

\$30.00



Arduino Projects for Amateur Radio

by Jack Purdum, Dennis Kidder

Boost Your Ham Radio's Capabilities Using Low Cost Arduino Microcontroller Boards

Do you want to increase the functionality and value of your ham radio without spending a lot of money? This book will show you how! *Arduino Projects for Amateur Radio* is filled with step-by-step microcontroller projects you can accomplish on your own — no programming experience necessary.

\$30.00



Make Your Own PCBs with EAGLE

by Eric Kleinert

Featuring detailed illustrations and step-by-step instructions, *Make Your Own PCBs with EAGLE* leads you through the process of designing a schematic and transforming it into a PCB layout. You'll then move on to fabrication via the generation of standard Gerber files for submission to a PCB manufacturing service. This practical guide offers an accessible, logical way to learn EAGLE and start producing PCBs as quickly as possible.

\$30.00



Build Your Own Transistor Radios

by Ronald Quan

A Hobbyist's Guide to High Performance and Low-Powered Radio Circuits

Create sophisticated transistor radios that are inexpensive yet highly efficient. Inside this book, it offers complete projects with detailed schematics and insights on how the radios were designed. Learn how to choose components, construct the different types of radios, and troubleshoot your work.

**Paperback, 496 pages*

\$49.95

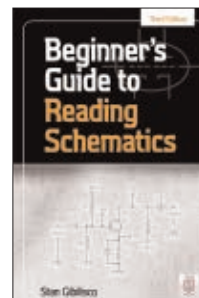


Beginner's Guide to Reading Schematics, 3E

by Stan Gibilisco

Navigate the roadmaps of simple electronic circuits and complex systems with help from an experienced engineer. With all-new art and demo circuits you can build, this hands-on, illustrated guide explains how to understand and create high-precision electronics diagrams. Find out how to identify parts and connections, decipher element ratings, and apply diagram-based information in your own projects.

\$25.00



How to Diagnose and Fix Everything Electronic

by Michael Jay Geier

Master the Art of Electronics Repair

In this hands-on guide, a lifelong electronics repair guru shares his tested techniques and invaluable insights.

How to Diagnose and Fix Everything

Electronic shows you how to repair and extend the life of all kinds of solid-state devices, from modern digital gadgetry to cherished analog products of yesteryear.

\$24.95



Programming PICs in Basic

by Chuck Hellebuyck

If you wanted to learn how to program microcontrollers, then you've found the right book! Microchip PIC microcontrollers are being designed into electronics throughout the world and none is more popular than the eight-pin version. Now the home hobbyist can create projects with these little microcontrollers using a low cost development tool called the CHIPAXE system and the Basic software language. Chuck Hellebuyck introduces how to use this development setup to build useful projects with an eight-pin PIC12F683 microcontroller.



\$14.95

Programming Arduino Next Steps: Going Further with Sketches

by Simon Monk

In this practical guide, electronics guru Simon Monk takes you under the hood of Arduino and reveals professional programming secrets. Also shows you how to use interrupts, manage memory, program for the Internet, maximize serial communications, perform digital signal processing, and much more. All of the 75+ example sketches featured in the book are available for download.



\$20.00

Order online @ www.store.nutsvolts.com
Or CALL 1-800-783-4624 today!

EDUCATIONAL

Beginners Guide Book Combo.



Only \$85.95
Plus
FREE Priority Mail Shipping
US Only

An Arduino Workshop

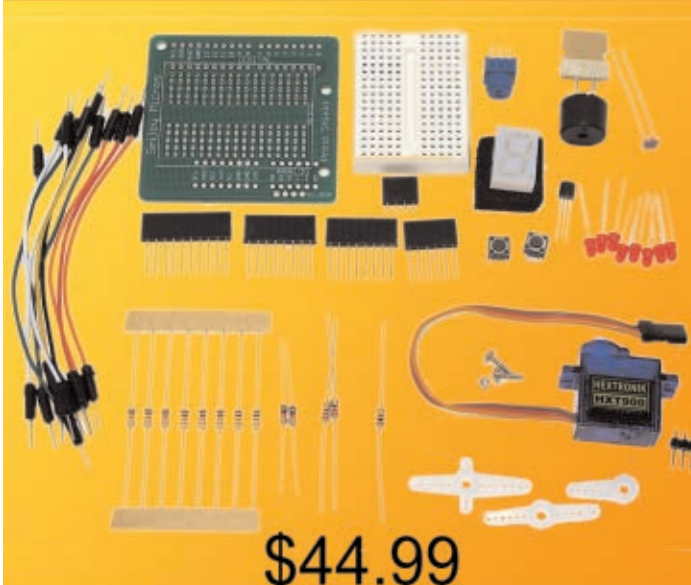
Are you puzzled about the Arduino but finding it difficult to get all the pieces in one place?



Joe Pardue
SmileyMicros.com

Book and Kit Combo \$124.95

Arduino Classroom Arduino 101 Projects Kit



From Smiley's Workshop

CD-ROM SPECIAL

Nuts & Volts 11 CD-ROMs & Hat Special!

That's 132 issues.
Complete with supporting
code and media files.



Free Shipping!

Only \$229.95
or \$24.95 each.



The Nuts & Volts Pocket Ref

All the info you need at your fingertips!

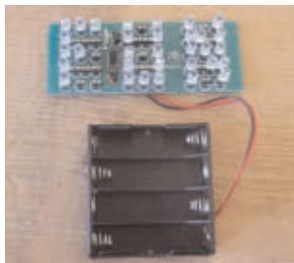
This great little book is a concise
all-purpose reference featuring
hundreds of tables, maps,
formulas, constants &
conversions.
AND it still fits in
your shirt pocket!

Only \$12.95

Visit <http://store.nutsvolts.com> or call (800) 783-4624

PROJECTS

No Nonsense Annunciator Kit

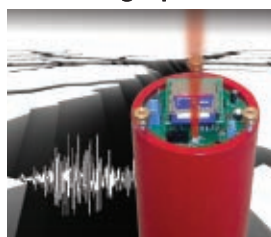


The no nonsense/no microprocessor annunciator is a great little circuit that helps you get your message out without spending too much money. Put two circuits together and you'll have a six letter annunciator!

This kit is also a fun project to refine your soldering skills with its 102 socket pin connection points. WOW, that's a lot of soldering!

\$19.95

Seismograph Kit

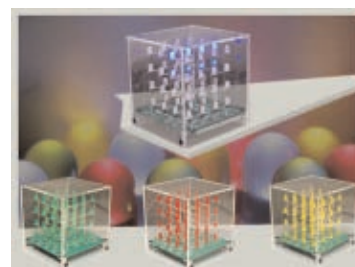


Now you can record your own shaking, rattling, and rolling.

The Poor Man's Seismograph is a great project/device to record any movement in an area where you normally shouldn't have any. The kit includes everything needed to build the seismograph. All you need is your PC, SD card, and to download the free software to view the seismic event graph.

\$79.95

3D LED Cube Kit



This kit shows you how to build a really cool 3D cube with a 4 x 4 x 4 monochromatic LED matrix which has a total of 64 LEDs. The preprogrammed microcontroller that includes 29 patterns that will automatically play with a runtime of approximately 6-1/2 minutes.

Colors available: Green, Red, Yellow & Blue

\$57.95

Solar Charge Controller Kit 2.0



If you charge batteries using solar panels, then you can't afford not to have them protected from over-charging. This 12 volt/12 amp charge controller is great protection for the money. It is simple to build, ideal for the novice, and no special tools are needed other than a soldering iron and a 9/64" drill!

\$27.95

Geiger Counter Kit



This kit is a great project for high school and university students. The unit detects and displays levels of radiation, and can detect and display dosage levels as low as one micro-roentgen/hr. The LND 712 tube in our kit is capable of measuring alpha, beta, and gamma particles.

Partial kits also available.

\$159.95

Super Detector Circuit Set



Pick a circuit!

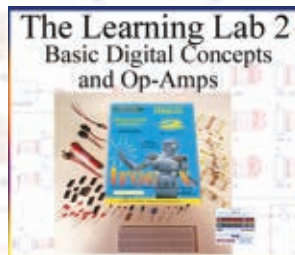
With one PCB you have the option of detecting wirelessly: temperature, vibration, light, sound, motion, normally open switch, normally closed switch, any varying resistor input, voltage input, mA input, and tilt, just to name a few.

\$32.95

FOR BEGINNER GEEKS!



\$59.95



\$49.95



\$39.95

These labs from LF Components show simple and interesting experiments and lessons, all done on a solderless circuit board.

As you do each experiment, you learn how basic components work in a circuit, and continue to build your arsenal of knowledge with each successive experiment.

For more info and lab details, please visit our webstore.

NEW PRODUCTS

Continued from page 47

The receiver architecture is that of a single conversion super-hetrodyne receiver that is capable of receiving AM, SSB, or CW.

The receiver incorporates a dual gate FET as an RF amplifier with manual peaking and gain controls. A ceramic filter is used in the IF section with a front panel switch that controls a broad or narrow IF response.

Other front panel controls include audio drive, beat frequency oscillator setting, and a band switch.

Even though the enclosure is quite unique, the performance of the receiver is not to be dismissed, with a 1 uV sensitivity minimum discernable signal being recognized.

For those who do not want a "meat can" enclosure or want additional receive frequency capability, check out National RF's new FB7-NX receiver, which is a professional grade miniature HF receiver with ancillary outputs (such as oscillator frequency output and muting functions) that make it suitable for integration with a transmitter and the amateur radio service.

For more information, contact:

National RF, Inc.

Web:

www.NationalRF.com

CLASSIFIEDS

SURPLUS

SURPLUS ELECTRONIC PARTS & ACCESSORIES



Over
20,000
Items
In
Stock

Belts
Cables
Connectors
Fans

Hardware
LEDs
Motors
Potentiometers

Relays
Semiconductors
Service Manuals
Speakers

Switches
Test Equipment
Tools
VCR Parts

Surplus Material Components
SMC ELECTRONICS
www.smcelectronics.com

No Minimum Order.
Credit Cards and PAYPAL Accepted.
Flat \$4.95 per order USA Shipping.

KITS/PLANS



QKITS LTD
sales@qkits.com

1 888 GO 4 KITS

Arduino • Raspberry Pi
Power Supplies
MG Chemicals
RFID

\$8.50
flat rate
shipping



Visit us at:
www.qkits.com

CONTROLLERS

Join The
**INTERNET of THINGS
REVOLUTION**



TRI SUPER PLCs

Powerful & Easy Ladder
+ BASIC Programming
Ethernet Integrated
MODBUS TCP/IP
DI/Os & AI/Os Integrated

tel : 1 877 TRI-PLCS
web : www.triplc.com/nv.htm



**TRIANGLE
RESEARCH
INTERNATIONAL**

MISC FOR SALE

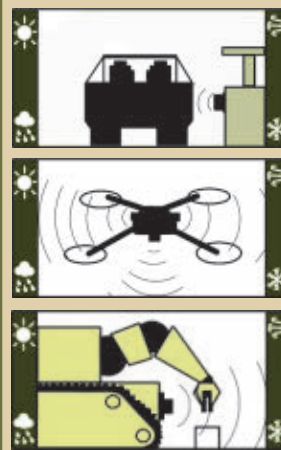
**SPEAKER BUSINESS CLOSED
AFTER 35 YEARS.**

SELLING \$250,000 INVENTORY OF
SPEAKERS, SPEAKER PARTS AND
MANUFACTURING EQUIPMENT AT A
FRACTION OF COST. ALL OFFERS
CONSIDERED. ISE, SAN BENITO, TX.
Info www.iseliquidator.com
956-444-0004, 888-351-5550

www.nutsvolts.com

ROBOTICS

Need sensors?



www.maxbotix.com

Like to
build robots?
Then, you need
a subscription
to **SERVO!**



www.servomagazine.com

HARDWARE WANTED

**DEC EQUIPMENT
WANTED!!!**

Digital Equipment Corp.
and compatibles.
Buy - Sell - Trade

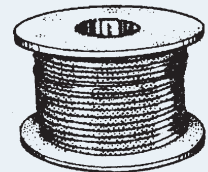
CALL KEYWAYS 937-847-2300
or email buyer@keyways.com

LIGHTING



www.ReactiveLighting.com

WIRE/CABLE



ANAHEIM WIRE, INC.
**Master distributor of
electrical and electronic
wire and cable since
1973.** Items available from
stock: Hook up wire, Shrink
tubing, Cable ties, Connectors,
etc. Wire cut & strip to specs,
twisting, stripping. If interested,
please call **1-800-626-
7540**, FAX: 714-563-8309.
Visa/MC/Amex. See us on
the Internet: www.anaheimwire.com
or email: info@anaheimwire.com.

SERVICES



Sandpiper Electronics Services LLC

"design of custom electronics instrumentation"

Circuit design & drawings, PCB layout, fabrication & population
Prototype development for research, industry and inventors.
39 years experience in electronics development for research &
flight applications, laser systems & laboratory research

FREE no obligation consultations

www.sandtronics.com

AUDIO/VIDEO

**PARTS
EXPRESS**
YOUR ELECTRONICS CONNECTION

**OVER 18,000
ELECTRONIC PARTS
IN STOCK**



Call or visit us online today to receive
your **FREE** copy of our 2015 catalog!

parts-express.com/nuts
1-800-338-0531

>>> QUESTIONS

LED vs. Incandescent Lamps

I have been tasked with the chore of replacing 300 watt incandescents (5900 lumen). How many of what kind of LEDs and current-limiting diodes in series/parallel do I need to fool the human eye into thinking it is seeing a brighter, more pleasant level of lumens?

**#4151 James McFadden
St. Maries, ID**

Test Lead Wire

Anyone know where I can purchase small quantities (25 ft rolls) of the different color jackets of good Beldon or (?) 65/30 test lead wire? I see it in 100 ft rolls \$\$, 10 colors, but that would be over a \$1,000 for all ten.

**#4152 Terry Arnall
Hayward, CA**

Over Current for PWM Circuit

I have a Marlin P. Jones DC motor speed controller (Part 31566MD, 6-24 volts, 20 amps max). I need to add an over current circuit to it. I inserted two 0.1 ohm/five watt resistors in series with the motor -lead and the M-connection on the controller. My scope displays a steady 0.6 volts DC level across it. The PWM waveform changes from 2 μ sec to 40 μ sec in length as the output of the speed controller is increased from 0 to 3 amps, while my DVM displays 0.02 VDC to 1.3 VDC for the same range of output.

So, the question is "What kind of circuit can I add across the resistors to get a VDC reading?" I have tried an NPN transistor, base lead to the motor

-lead, and the emitter lead to the M-connection. (With a 10K collector resistor to +12 VDC.) The collector voltage went from +12 volts to +3 volts as the controller output went from 0 to 3 amps.

Next, I connected the collector voltage to an LM324N quad op-amp set up as a voltage comparator. The +lead of an LM324 went to a 200K pot, connected between +12 VDC and Gnd. The transistor's output went to the -input of the same op-amp. A 1M ohm resistor is connected from output to +input for hysteresis. (This output should go high to set a CD4013N flip-flop at an over current condition.)

The problem is that the output of the op-amp's output does NOT change at the point when the voltage at the +input is greater than the -input. The op-amp's output changes as the voltage from the transistor decreases. I used the LM324N quad op-amp because it has four op-amps in one chip, and it works with a single +12 VDC supply. It would be helpful if the new circuit could use it also, but not necessary. I could use a PIC16F628 or an Arduino Nano, if you design with them.

**#4153 Patrick Fleming
Hoffman Estates, IL**

>>> ANSWERS

[#12145 - December 2014]

False Readings

I bought one of those Internet-aware soil moisture devices a few months ago. It worked great at first, but now the electrodes are oxidized and

the readings are falsely dry because of the increased resistance. Scrubbing the electrodes with steel wool works for about a week. Any ideas for a permanent solution?

If the sensor is powered with a direct current voltage, you may be out of luck. However, here are some possible 'fixes.'

I read that the phone company, many years ago, had a problem with corrosion and switched from a negative ground to a positive ground which helped to solve their corrosion problem. If you could insure the system is + grounded, this may help.

I think that most serious outside systems use AC sensors, and if they use DC, employ a positive ground and are not powered unless a measurement is needed.

**Marc Forgey
Seattle, WA**

[#3151 - March 2015] X10 Cable Build

I have an early X10 Home Control Timer (Model CP-290) to which I have lost the programming cable. Does anyone know where I can get the pinout so I could fashion my own replacement cable?

The CP-290 is very old and was sold in the early 1980s. The communication protocol is RS232 serial.

Baud Rate: 600
Data Bits: 8
Parity: None
Stop Bits: 1

The connector on the back is a standard five-pin DIN socket. Looking at the back of the unit, the pins are 5-4-3-2-1, starting at the left and going counter-clockwise to the right with pin 5 on the left, 3 on the bottom, and 1 on the right.

Pin 1 - No Connection
Pin 2 - Receive Data (In)

All questions AND answers are submitted by *Nuts & Volts* readers and are intended to promote the exchange of ideas and provide assistance for solving technical problems. All submissions are subject to editing and will be published on a space available basis if deemed suitable by the publisher. Answers are submitted

by readers and **NO GUARANTEES WHATSOEVER** are made by the publisher. The implementation of any answer printed in this column may require varying degrees of technical experience and should only be attempted by qualified individuals.

Always use common sense and good judgment!

Send all questions and answers by email to forum@nutsvolts.com
or via the online form at www.nutsvolts.com/tech-forum

Pin 3 - Ground
Pin 4 - Transmit Data (Out)
Pin 5 - No Connection

To plug into a computer like an IBM compatible these connections would go to a nine-pin female serial connector:

CP290 → DE9
2 RxD → 3 TxD
3 Gnd → 5 Gnd
4 TxD → 2 RxD

Rick Swenton
via email

[#3152 - March 2015] Lightning Protector

In a recent thunderstorm, a nearby lightning strike took out some of the electronics at my neighbor's house. Is there anything a DIYer like me can build to protect my delicate electronics – other than unplugging everything? Something with MOVs maybe?

#1 Lightning protection is a complex issue, including home entrance cable protection, bonding of large metallic structures, and even grounding of rain gutters and downspouts. A good place to start is a free PDF from www.lightningsafety.com/nlsi_lhm/IEEE_Guide.pdf.

Because a lightning bolt can pack a 500 MJ wallop – far beyond any MOV rating – surge protectors are not protection against a direct strike, but they do help limit inductive surges (e.g. lightning hitting a tree nearby, inducing a current in house wiring). In brief, running a hefty ground wire from gutters, external antennas, etc., to an effective ground in conductive soil is the first line of defense.

MOV surge suppressors have saved my PC and appliances from one damaging surge – as shown by blown internal fuses and smoking MOVs!

B. Bresnik
via email

#2 The simplest solution to lightning-induced surge protection is to use a commercially available surge-protected outlet strip. There are numerous sources for these items, and you may even find a suitable device at your local hardware store.

The important thing to understand is that a lightning strike conducts huge oscillatory currents. A varying electrical current will generate a changing magnetic field which, in turn, will induce superimposed voltages in nearby conductors – including service drops from the utility pole to your house (e.g., electrical power, TV/Internet cable, and telephone). Such surges can be induced both line-to-line and line-to-ground in the electrical power service drop (and for balanced-line applications such as telephone). Properly designed surge suppressors provide both line-to-ground and line-to-line protection for such circuits.

Surge voltages induced line-to-ground arise because such devices often are connected to more than one source of surge voltage. For example, your television set is connected to utility power and also connected to the TV signal cable. Likewise, your computer may be connected to utility power, to a cable from your Internet service provider, and to a telephone cable (for fax service). Unless these cables/wires are all run together throughout the house (and this practice is discouraged due to the possibility of capacitive cross-coupling), one or more loops exist, and within each loop, the surge voltage induced by the lightning strike is a direct function of the areas enclosed by the loop.

It follows that effective surge suppression can only be accomplished by feeding all of the incoming electrical services through what is called a "surge-protective window." In such a structure, surge suppression elements such as metal

oxide varistors (MOVs), avalanche diodes, or gas tubes can clamp impulse voltages to a common reference point plane which, in turn, is connected to earth ground. This can be effected by using a surge-protected outlet strip that also incorporates protection for telephone and cable lines. Typical examples of this all-inclusive surge protection are devices available from Belkin (e.g., www.belkin.com/us/BV112234-08-Belkin/p/P-BV112234-08/) and Tripp-Lite (e.g., www.tripplite.com/av-home-theater-surge-protector-isobar-10-outlets-8-ft-cord-3240-joule-3-line-coax-ethernet-tel-network~AVBAR10/).

All cables exiting the surge suppressor block should be run together wherever possible, secured periodically by twist-ties or other means. This method ensures that negligible induction areas exist into which surge voltages can be introduced. Capacitive-coupled interactions are no longer a problem because any prior surge voltages have already been stripped from the cables by the surge suppressor block.

Please note that the protectors identified above are "Cadillacs" because they provide surge suppression for all common power and media transport wiring. Sometimes just a simple one-outlet surge suppressor will do the job – or, for example, a single-outlet suppressor with built-in telephone line surge suppression – both at significantly lower cost. I even use single-outlet surge suppressors to protect my coffee maker and washing machine because each of these devices contains electronic modules that are expensive to repair.

The important consideration is to maintain the "surge protective window" approach to the problem as outlined above.

Peter A. Goodwin
Rockport, MA

For reproduction parts like rubber, knobs, dial faces, etc., and other cosmetic parts, I recommend Renovated Radios (www.renovatedradios.com).

J.W. Koebel

Tread Thread

I have a minor correction for Theron Wierenga's January 2014 barn door tracker. The article states that the tread used to mount most cameras is 1/4-28; it's actually 1/4-20.

Arlen Raasch

Arlen, thank you for your comments. I concur and also appreciate the heads-up on the typo about the thread size.

Theron Wierenga

Loves Ham

I absolutely love the addition of the ham radio centric articles in the latest volumes of *Nuts & Volts*. I think it's a great way to expose readers to the technical aspects of ham radio, of which they may not know about. Keep up the great work!

73, Ryan Clarke KJ6MSG

Spaced Out on CubeSats

There are several errors and omissions in Paul Verhage's February 2015 Near Space column on CubeSats. The Pumpkin standard board is NOT a PC-104 board. The form factor is a PC-104 size and shape and the 104-pin connector is the same, but the signals on that are entirely different.

If you plug a PC-104 board into a Pumpkin or similar board, you will certainly burn up both.

He listed four companies producing subsystems for CubeSats

— the ones with an easy-to-find website. There are at least 20 that produce various modules and more every day.

Paul discussed CubeSat air frames made from panels or skeletonized. There are any number of ways to make the structure of a CubeSat, and many builders make their own (often customized), so their particular sensor or other subsystem will fit.

There is no end to the creativity builders have engaged in for structures. For example, the PrintSat structure is entirely 3D printed.

Paul also mentioned expansion and contraction due to temperature changes as a reason solar panels are "clipped" to a structure. That's not the reason for the clips at all. It's for ease of assembly and disassembly.

While the thermal coefficients of solar cells and aluminum are different, over the space of 10 cm, there isn't enough difference to create risk for the cells. More often, the substrate the cells are on is bonded to or isolated from the structure to allow heat from the cells to be conveyed to the structure, or to keep that heat away from the structure because a particular satellite's thermal budget needs more heat or less.

There are numerous ways solar panels are connected to structures; clips are a minor player in that game, with nuts and bolts being much more prevalent.

Li-Po and Li batteries used in CubeSats were discussed. In reality, virtually every battery chemistry has been and continues to be used. The choice depends on the power needs. For example, NiCd's are still used when a large amount of current is needed for a short time.

The only constraints on chemistry are what the launch vehicle provider will allow. For the most part, they will allow anything because the CubeSat is in a closed deployer.

Regarding the P-POD deployer from Cal Poly. There are at least five companies making deployers now, with more being invented every day. Two of the more popular are the ISIS and Planetary Systems, Inc. NASA Ames makes its own: the NLAS.

It was indicated that a CubeSat radio only needs to transmit at a couple of watts to be heard with a handheld amateur radio. The threshold is closer to 100 mw for FM voice, and less than that for CW.

I was stated that Doppler correction is necessary. On VHF with FM, it isn't. The Doppler shift is only about 1.2 kHz for a LEO satellite, and with FM it's unnecessary to correct for that. It is, of course, for SSB.

Jim White

Colorado Satellite Services, LLC

Thanks for the information on the Pumpkin board. It does have the appearance and measurements of a PC-104 board. Therefore, it would be important to know the function of each pin so you don't destroy boards by stacking them together.

As with many products, it's important to read the datasheets. I guess it goes to show that what looks like a standard might not be in all cases.

I read how NiCd's are space qualified, but wasn't aware they were still heavily used.

I looked at CubeSats for the first time several years ago. Since then, it has indeed grown. I get the feeling that I'm watching CubeSats grow like the home PC grew several decades ago.

It would seem to me that beginners would be well served looking at CubeSat kits first and then expanding out from there.

Paul Verhage

AMATEUR RADIO

AND TV

National RF23
Ramsey Electronics82-83

BUYING ELECTRONIC SURPLUS

Earth Computer Technologies68
Weirdstuff Warehouse23

CCD CAMERAS/VIDEO

Ramsey Electronics82-83

CIRCUIT BOARDS

Accutrace Inc..15
AP Circuits64
Dimension Engineering6
ExpressPCB37
Front Panel Express LLC64
Saelig Co. Inc.14

COMPONENTS

LF Components.....69
MaxbotixBack Cover
SDP/SI23

COMPUTER

Hardware

Earth Computer Technologies68
Weirdstuff Warehouse23

Microcontrollers /

I/O Boards

MikroElektronika3
Technologic Systems15

DESIGN/ENGINEERING/ REPAIR SERVICES

Accutrace Inc..15
Cleveland Institute of Electronics..14
ExpressPCB37
Front Panel Express LLC64
National RF23

DRIVE COMPONENT

CATALOGS

SDP/SI23

EDUCATION

Boxed Kit Amps23
Cleveland Institute of Electronics..14
Command Productions7
LF Components.....69
NKC Electronics23
Poscope37

EMBEDDED TOOLS

NetBurner2

ENCLOSURES

Front Panel Express LLC64

EVENTS

Maker Faire65
RoboGames79

HI-FI AUDIO

Boxed Kit Amps23

KITS & PLANS

Boxed Kit Amps23
Earth Computer Technologies68
LF Components.....69
NetBurner2
NKC Electronics23
Ramsey Electronics82-83

MISC./SURPLUS

All Electronics Corp.46
Front Panel Express LLC64
Weirdstuff Warehouse23

MOTORS

Servo City/Robot Zone5

PCB ASSEMBLY

Accutrace Inc..15

PROGRAMMERS

MikroElektronika3

RF TRANSMITTERS/ RECEIVERS

National RF23

ROBOTICS

Cleveland Institute of Electronics..14
Lemos International Co.61
MaxbotixBack Cover
SDP/SI23
Servo City/Robot Zone5

SATELLITE

Lemos International Co.61
Team Synergy Moon57

SENSORS

MaxbotixBack Cover

TEST EQUIPMENT

Dimension Engineering6
NKC Electronics23
Poscope37
Saelig Co. Inc.14

TOOLS

MikroElektronika3
NetBurner2
PanaVise15
Poscope37

WIRE, CABLE AND CONNECTORS

Servo City/Robot Zone5

Accutrace Inc..15
All Electronics Corp.46
AP Circuits64
Boxed Kit Amps23
Cleveland Institute of
Electronics14
Command Productions7
Dimension Engineering6
Earth Computer Technologies 68
ExpressPCB37
Front Panel Express LLC64
Lemos International Co.61
LF Components.....69
Maker Faire65
MaxbotixBack Cover
MikroElektronika3
National RF23
NetBurner2
NKC Electronics23
PanaVise15
Poscope.....37
Ramsey Electronics82-83
Saelig Co. Inc.14
SDP/SI23
Servo City/Robot Zone5
Team Synergy Moon57
Technologic Systems15
Weirdstuff Warehouse23



Super-Pro FM Stereo Radio Station

SPRING SALE!

- ✓ PLL synthesized for drift-free operation
- ✓ Built-in mixer - 2 line inputs and one microphone input, line level monitor output!
- ✓ Frequency range 88.0 to 108.0, 100 kHz steps
- ✓ Precision active low-pass "brick wall" audio filter!
- ✓ Dual LED bar graph audio level meters!
- ✓ Automatic adjustable microphone ducking!
- ✓ Easy to build through-hole design!

The true professional workhorse of our FM Stereo transmitter line, the FM100B has become the transmitter of choice for both amateurs and professionals around the world. From the serious hobbyist to churches, drive-in theaters, colleges and schools, it continues to be the leader. Not just a transmitter, the FM100B is a fully functional radio station and provides everything but the audio input and antenna system! Just add that and you're on the air!

This professional synthesized transmitter is adjustable directly from the front panel with a large LED digital read-out of the operating frequency. Just enter the setup mode and set your frequency. Once selected and locked you are assured of a rock stable carrier with zero drift. The power output is continuously adjustable throughout the power range of the model selected. In addition, a new layer of anti-static protection for the final RF amplifier stage and audio inputs has been added to protect you from sudden static and power surges.

Audio quality is equally impressive. A precision active low-pass brick wall audio filter and peak level limiters give your signal maximum "punch" while preventing overmodulation. Two sets of rear panel stereo line level inputs are provided with front panel level control for both. Standard unbalanced "RCA" line inputs are used to make it simple to connect to the audio output of your computer, MP3 player, DVD player, cassette deck or any other consumer audio source. Get even more creative and use our K8094 for digital storage and playback of short announcements and ID's! In addition to the line level inputs, there is a separate front panel microphone input.

All three inputs have independent level controls, eliminating the need for a separate audio mixer! Just pot-up the source control when ready, and cross fade to the 2nd line input or mic! It's that simple! In addition to the dual stereo line inputs, a stereo monitor output is provided. This is perfect to drive studio monitors or local in-house PA systems. The FM100B series includes an attractive metal case, whip antenna and built-in 110/220VAC power supply. A BNC connector is also provided for an external antenna. Check out our Tru-Match FM antenna kit, for the perfect mate to the FM100B transmitter. We also offer a high power kit as well as an export-only assembled version that provides a variable RF output power up to 1 watt. The 1 watt unit must utilize an external antenna properly matched to the operating frequency to maintain a proper VSWR to protect the transmitter.

(Note: The FM100B and FM100BEX are do-it-yourself learning kits that you assemble. The end user is responsible for complying with all FCC rules & regulations within the US or any regulations of their respective governing body. The FM100BWT is for export use and can only be shipped to locations outside the continental US, valid APO/FPO addresses or valid customs brokers for documented end delivery outside the continental US.)

FM100B Super-Pro FM Stereo Radio Station Kit, 5uW to 25mW Output

FM100BEX Super-Pro FM Stereo Radio Station Kit, 5uW to 1W Output

\$239.95
\$299.95

Tru-Match FM Antenna

SPRING SALE!

The unique PVC waterproof design has made the TM100 one of the most reliable antennas available for the private broadcast-er! Broadband over the entire 88-108MHz range makes installation a breeze without any tuning. Easy "F" connector termination.



TM100 Tru-Match FM Antenna Kit

\$59.95

FM Low Pass Filter

SPRING SALE!

Customers have complained about "dirty" imported FM broadcasters and exciters, so we developed this low pass filter based on our PX50 broadcast xmtr. Really cleans up your output, and connects easily between your transmitter & antenna. Maximum input power is 50 watts.



FMLP1 FM Low Pass Filter Kit

\$25.95

Tunable FM Stereo Transmitter Kit

SPRING SALE!

Here is the famous entry-level kit that will teach the basics of FM Broadcast transmission while finding many uses around the home or dorm room.



The FM10C has plenty of power to cover your home, backyard, or city block and tunes through the entire 88-108MHz band in three separate ranges with a tuned LC circuit. Runs on a 9V battery or optional AC power adapter.

(Note: The FM10C is a do-it-yourself learning kit that you assemble. Please remember that the end user is responsible for complying with all FCC rules & regulations within the US, or any regulations of their respective governing body in regards to the application and use of the FM10C.)

FM10C FM Stereo Transmitter Kit

\$39.95

Collinear Vertical FM Antenna

SPRING SALE!

Our 5/8 wave omni antenna has been the standard for LPFM installations worldwide. Provides 3.4dB gain while keeping the signal radiation low to the horizon for maximum range. Field tuneable over the entire FM range for a perfect match. SO239 connector for PL259 plug.



FMA200E Omnidirectional FM Antenna

\$119.95

Synthesized FM Stereo Transmitter Kit

SPRING SALE!

The FM25 has been the hobbyist's standard for FM synthesized stereo transmitters for more than two decades!



Just plug in the stereo left/right audio from your MP3 player, CD player, or computer, and broadcast it on any frequency in the standard FM broadcast band.

The sound quality and stereo separation of this little transmitter will keep the pickiest audiophile happy. The FM25B features a PIC microprocessor for easy frequency programming through board mounted DIP switches. The transmit frequency is Phase Locked Loop (PLL) controlled for unparalleled stability making frequency drift a thing of the past, extremely critical for digital tuners. The RF output level is adjustable from 5uW to 25 mW via a potentiometer. Use the built-in whip antenna or use an external antenna with the standard "F" external antenna connector on the rear panel. Just plug it in and you're on the air.

(Note: The FM25B is a do-it-yourself learning kit that you assemble. Please remember that the end user is responsible for complying with all FCC rules & regulations within the US, or any regulations of their respective governing body in regards to the application and use of the FM25B.)

FM25B Synth FM Stereo Xmtr Kit

\$119.95

Classic Nixie Tube Clocks



Our next generation of classic Nixie tube clocks perfectly mesh today's technology with the Nixie era technology of the 60's. Of course, features you'd expect with a typical clock are all supported with the Nixie clock... and a whole lot more!

The clocks are programmable for 12 or 24 hour mode, various AM/PM indications, programmable leading zero blanking, and include a programmable alarm with snooze as well as date display, 4 or 6 tube, kit or assembled!

We then jumped the technological time line of the 60's Nixie displays by adding the latest multi-colored LEDs to the base of the Nixie tubes to provide hundreds of illumination colors to highlight the glass tubes! The LED lighting can be programmed to any color and brightness combination of the colors red, green, or blue to suit your mood or environment.

Then we leaped over the technological time line by integrating an optional GPS time base reference for the ultimate in clock accuracy! The small optional GPS receiver module is factory assembled and tested, and plugs directly into the back of the clock to give your Nixie clock accuracy you could only dream of!

The clocks are available in our signature hand rubbed Teak & Maple, or futuristic clear acrylic bases. You also have your choice of IN-14 or highly sought after IN-8-2 nixie tubes (for the 6-tube clock).

NIXIE Classic Nixie Tube Clock Kits From \$229.95

Air Blasting Ion Generator

Generates negative ions along with a hefty blast of fresh air, all without any noise! The steady state DC voltage generates 7.5kV DC negative at 400uA, and that's LOTS of ions! Includes 7 wind tubes for max air! Runs on 12-15VDC.



IG7 Ion Generator Kit

\$64.95

HV Plasma Generator

Generate 2" sparks to a handheld screwdriver! Light fluorescent tubes without wires! This plasma generator creates up to 25kV at 20kHz from a solid state circuit! Build plasma bulbs from regular bulbs and more! Runs on 16VAC or 5-24VDC.



PG13 HV Plasma Generator Kit

\$64.95

Signal Magnet Antenna

The impossible AM radio antenna that pulls in the stations and removes the noise, interference, and static crashes from your radio! Also helps that pesky HD AM Radio stay locked! Also available factory assembled.



SM100 Signal Magnet Antenna Kit

\$89.95

Broadband RF Preamp

Need to "perk-up" your counter or other equipment to read weak signals? This preamp has low noise and yet provides 25dB gain from 1MHz to well over 1GHz. Output can reach 100mW! Runs on 12 volts AC or DC or the included 110VAC PS. Asmb.



PR2 Broadband RF Preamp

\$69.95

Active Receive Antenna

The popular antenna for the serious DX'ers works on all bands - shortwave, HF, VHF, and UHF yet performs like a 60' long wire antenna! Provides over 15dB of gain, and includes auto-off RF bypass and front panel gain control.



AA7C Active Antenna Kit

\$59.95



There's only so much room on these two pages, so check it all out in our new virtual electronic catalog! Flip through the pages and search with ease! Visit www.ramseycatalog.com

Follow Us and SAVE \$\$

Follow us on your favorite network site and look for a lot of super deals posted frequently... exclusively for our followers!



Digital Controlled FM Stereo Transmitter

- ✓ PLL synthesized for drift free operation
- ✓ Front panel digital control and display of all settings and parameters!
- ✓ Professional metal case for noise-free operation
- ✓ EMI filtering on audio and power inputs
- ✓ Super audio quality, rivals commercial broadcasts
- ✓ Available in domestic kit or factory assembled export versions



For more than a decade we've been the leader in hobbyist FM radio transmitters. We told our engineers we wanted a new technology transmitter that would provide FM100 series quality without the advanced mixer features. They took it as a challenge and designed not one, but TWO transmitters!

The FM30B is designed using through-hole technology and components and is available only as a do-it-yourself kit with a 25mW output very similar to our FM25 series. Then the engineers redesigned their brand-new design using surface mount technology (SMT) for a very special factory assembled and tested FM35BWT version with 1W output for our export only market! All settings can be changed without taking the cover off! Enter the setup mode from the front panel and step through the menu to make all of your adjustments. A two line LCD display shows you all the settings! In addition to the LCD display, a front panel LED indicates PLL lock so you know you are transmitting.

Besides frequency selection, front panel control and display gives you 256 steps of audio volume (left and right combined) as well as RF output power. A separate balance setting compensates for left/right differences in audio level. In addition to settings, the LCD display shows you "Quality of Signal" to help you set your levels for optimum sound quality. And of course, all settings are stored in non-volatile memory for future use! Both the FM30B and FM35BWT operate on 13.8 to 16VDC and include a 15VDC plug-in power supply.

(Note: After assembly of this do-it-yourself hobby kit, the user is responsible for complying with all FCC rules & regulations within the US, or any regulations of their respective governing body. FM35BWT is for export use and can only be shipped to locations outside the continental US or valid APO/FPO addresses or valid customs brokers for end delivery outside the continental US.)

FM30B Digital FM Stereo Transmitter Kit, 0-25mW
FM35BWT Digital FM Stereo Transmitter, Assembled, 0-1W (Export ONLY)

SPRING SALE!
 \$169.95
 \$259.95

Electrocardiogram ECG Heart Monitor

- ✓ Visible and audible display of your heart rhythm!
- ✓ Bright LED "Beat" indicator for easy viewing!
- ✓ Re-usable hospital grade sensors included!
- ✓ Monitor output for professional scope display
- ✓ Simple and safe 9V battery operation

Use the ECG1C to astound your physician with your knowledge of ECG/EKG systems. Enjoy learning about the inner workings of the heart while, at the same time, covering the stage-by-stage electronic circuit theory used in the kit to monitor it. The three probe wire pick-ups allow for easy application and experimentation without the cumbersome harness normally associated with ECG monitors.

The documentation with the ECG1C covers everything from the circuit description of the kit to the circuit description of the heart! Multiple "beat" indicators include a bright front panel LED that flashes with the actions of the heart along with an adjustable level audio speaker output that supports both mono and stereo hook-ups. In addition, a monitor output is provided to connect to any standard oscilloscope to view the traditional style ECG/EKG waveforms just like you see in a real ER or on one of the medical TV shows! The fully adjustable gain control on the front panel allows the user to custom tune the differential signal picked up by the probes giving you a perfect reading and display every time! Additional patches are available in 10-packs. Operates on a standard 9VDC battery (not included) for safe and simple operation. Intended for hobbyist usage only. If you experience any cardiac symptoms, seek proper medical help immediately!

ECG1C Electrocardiogram Heart Monitor Kit With Case & Patches
ECG1WT Electrocardiogram Heart Monitor, Factory Assembled & Tested
ECGP10 Electrocardiogram Re-Usable Probe Patches, 10-Pack

\$44.95
 \$89.95
 \$4.95

Tickle-Stick Shocker

The kit has a pulsing 80 volt tickle output and a mischievous blinking LED. And who can resist a blinking light and an unlabeled switch! Great fun for your desk, "Hey, I told you not to touch!" Runs on 3-6 VDC.

TS4 Tickle Stick Kit \$9.95

Passive Aircraft Monitor **PATENTED!**

The hit of the decade! Our patented receiver hears the entire aircraft band without any tuning! Passive design has no LO, therefore can be used on board aircraft! Perfect for air-shows, hears the active traffic as it happens! Available kit or factory assembled.

ABM1 Passive Aircraft Receiver Kit \$89.95

Electret Condenser Mic

This extremely sensitive 3/8" mic has a built-in FET preamplifier! It's a great replacement mic, or a perfect answer to add a mic to your project. Powered by 3-15VDC, and we even include coupling cap and a current limiting resistor! Extremely popular!

MC1 Mini Electret Condenser Mic Kit \$3.95

12VDC Regulated Switching Supply

Go green with our new 12VDC 1A regulated supply. Worldwide input 100-240VAC with a Level-V efficiency! It gets even better, includes DUAL ferrite cores for RF and EMI suppression. All this at a 10 buck odd wallwart price! What a deal!

AC121 12VDC 1A Regulated Supply \$9.95

12VDC Worldwide Supply

It gets even better than our AC121 above! Now, take the regulated Level-V green supply, bump the current up to 1.25A, and include multiple blades for global country compatibility! Dual ferrite cores!

P529 12VDC 1.25A Global Power Supply \$19.95

Sniff-It RF Detector Probe

Measure RF with your standard DMM or VOM! This extremely sensitive RF detector probe connects to any voltmeter and allows you to measure RF from 100kHz to over 1GHz! So sensitive it can be used as a RF field strength meter!

RF1 Sniff-It RF Detector Probe Kit \$27.95

Electronic Chirping Cricket Sensor

Electronic cricket? Sounds just like those little black critters that seem to come from nowhere and annoy you with their chirp-chirp! And just like the little critters, we made it sensitive to temperature so when it gets warmer, it chirps faster!

That's right, you can even figure out the temperature by the number of chirps it generates! Just count the number of chirps over a 15 second interval, add 40, and you have the temperature in degrees Fahrenheit! Not as fancy as a digital thermometer but much more unique! And unlike its little black predecessor, the ECS1 operates from around 50°F to 90°F! I don't think there are too many real crickets chirping away at 90°F!

A unique thermistor circuit drives a few 555 ICs providing a variable chirp that is guaranteed to annoy everyone around you! But just watch their faces when you tell them the temperature outside! Runs on 9-12VDC or a standard 9V battery (not included).

Includes everything shown, including the speaker and battery clip, to make your cricket project a breeze.

ECS1 Electronic Cricket Sensor Kit \$24.95

The Learning Center!



Fun Electronic Learning Labs

- ✓ Learn and build!
- ✓ 130, 200, 300, & 500 in one labs!
- ✓ Practical through hole and SMT soldering labs!
- ✓ Integrated circuit AM/FM radio lab!
- ✓ Super comprehensive training manuals!

Starting out our "All in One" series, the PL130A, gives you 130 different electronic projects, together with a comprehensive 162 page learning manual. A great start for the kids...young and old! Next, check out the PL200, that gives you 200 very creative and fun projects, and includes a neat interactive front panel with 2 controls, speaker, LED display and a meter. From there, step up to our PL300, which gives you 300 separate electronic projects along with s 165 page learning and theory manual. The PL300 walks you through the learning phase of digital electronics. If you're looking for the ultimate lab kit, check out our PL500. It includes a whopping 500 separate projects, a 152 page starter course manual, a 78 page advanced course manual, and a 140 page programming course manual! The PL500 covers everything from the basics to digital programming!

If you are looking to either learn or hone up on your through hole or SMT soldering skills check our SP1A and SM200K Practical Soldering Labs. You will be a soldering master in no time!

We make it easy to learn IC's while at the same time, building a neat AM/FM radio with our AMFM108K AM/FM IC lab kit. You will have a blast AND learn!

PL130A	130-In-One Lab Kit	\$39.95
PL200	200-In-One Lab Kit	\$84.95
PL300	300-In-One Lab Kit	\$109.95
PL500	500-In-One Lab Kit	\$249.95
SP1A	Through Hold Soldering Lab	\$9.95
SM200K	SMT Practical Soldering Lab	\$22.95
AMFM108K	AM/FM IC Lab Kit & Course	\$36.95

GET THE INUTS and VOLTS DISCOUNT!

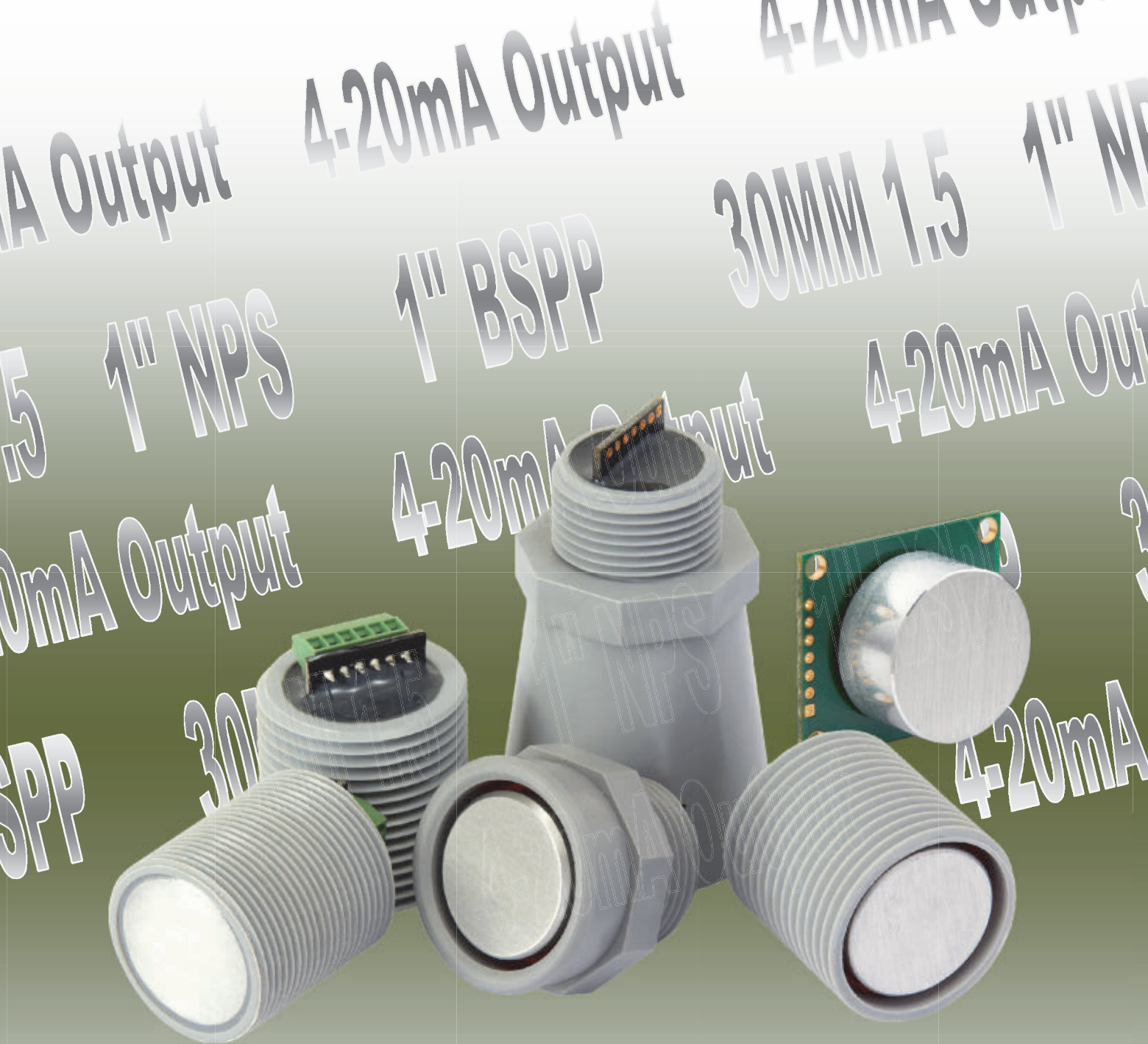
Mention or enter the coupon code **NVRMZ142** and receive 10% off your order!

800-446-2295
www.ramseykits.com

Prices, availability, and specifications are subject to change. We are not responsible for typos, stupid, printer's bleed, or confusion that April showers bring May flowers! Robin thinks winter is over, just because she lives in CA! Wrong! Visit www.ramseykits.com for the latest pricing, specials, terms and conditions. Copyright 2015 Ramsey Electronics®... so there!

RAMSEY ELECTRONICS®

590 Fishers Station Drive
 Victor, NY 14564
 (800) 446-2295
 (585) 924-4560



Multiple new options available for our
IP67 ultrasonic sensors

MaxBotix[®]
High Performance Ultrasonic Rangefinders



CE RoHS

www.maxbotix.com

info@maxbotix.com

